Efficient development of human language technology resources for resource-scarce languages

MJ Puttkammer

11313099

Thesis submitted for the degree *Doctor Philosophiae* in Linguistics and Literary Theory at the Potchefstroom Campus of the North-West University

Promoter: Prof GB van Huyssteen

Co-promoter: Prof E Barnard

September 2014

Abstract

The development of linguistic data, especially annotated corpora, is imperative for the human language technology enablement of any language. The annotation process is, however, often time-consuming and expensive. As such, various projects make use of several strategies to expedite the development of human language technology resources. For resource-scarce languages – those with limited resources, finances and expertise – the efficiency of these strategies has not been conclusively established. This study investigates the efficiency of some of these strategies in the development of resources for resource-scarce languages, in order to provide recommendations for future projects facing decisions regarding which strategies they should implement.

For all experiments, Afrikaans is used as an example of a resource-scarce language. Two tasks, viz. lemmatisation of text data and orthographic transcription of audio data, are evaluated in terms of quality and in terms of the time required to perform the task. The main focus of the study is on the skill level of the annotators, software environments which aim to improve the quality and time needed to perform annotations, and whether it is beneficial to annotate more data, or to increase the quality of the data. We outline and conduct systematic experiments on each of the three focus areas in order to determine the efficiency of each.

First, we investigated the influence of a respondent's skill level on data annotation by using untrained, sourced respondents for annotation of linguistic data for Afrikaans. We compared data annotated by experts, novices and laymen. From the results it was evident that the experts outperformed the non-experts on both tasks, and that the differences in performance were statistically significant.

Next, we investigated the effect of software environments on data annotation to determine the benefits of using tailor-made software as opposed to general-purpose or domain-specific software. The comparison showed that, for these two specific projects, it was beneficial in terms of time and quality to use tailor-made software rather than domain-specific or general-purpose software. However, in the context of linguistic annotation of data for resource-scarce languages, the additional time needed to develop tailor-made software is not justified by the savings in annotation time.

Finally, we compared systems trained with data of varying levels of quality and quantity, to determine the impact of quality versus quantity on the performance of systems. When comparing systems trained with gold standard data to systems trained with more data containing a low level of errors, the systems trained with the erroneous data were statistically significantly better. Thus, we conclude that it is more beneficial to focus on the quantity rather than on the quality of training data.

Based on the results and analyses of the experiments, we offer some recommendations regarding which of the methods should be implemented in practice. For a project aiming to develop gold standard data, the highest quality annotations can be obtained by using experts to double-blind annotate data in tailor-made software (if provided for in the budget or if the development time can be justified by the savings in annotation time). For a project that aims to develop a core technology, experts or trained novices should be used to single-annotate data in tailor-made software (if provided for in the budget or if the development time can be justified by the savings in annotation time).

Keywords:

Afrikaans; Automatic speech recognition; Lemmatisation; Resource-scarce languages; Human language technology; Resource development.

Abstract

Opsomming

Die ontwikkeling van linguistiese data, veral geannoteerde korpora, is van kardinale belang vir die ontwikkeling van mensetaaltegnologieë vir enige taal. Die annotasieproses is egter dikwels tydrowend en duur, en derhalwe maak verskeie projekte van verskillende strategieë gebruik om die ontwikkeling van mensetaaltegnologiehulpbronne te bespoedig. Vir hulpbronskaars tale – dié met beperkte talige bronne, finansies en kundigheid – is die doeltreffendheid van sommige van hierdie strategieë nog nie onomwonde bewys nie. Hierdie studie ondersoek die doeltreffendheid van sommige van hierdie strategieë in die ontwikkeling van hulpbronne vir hulpbronskaars tale ten einde aanbevelings vir toekomstige projekte te maak.

Vir al die eksperimente word Afrikaans as ’n voorbeeld van ’n hulpbronskaars taal gebruik. Twee take, naamlik lemma-identifisering van teksdata en ortografiese transkripsie van oudiodata, word volgens kwaliteit en die tyd wat dit neem om die taak te voltooi, geëvalueer. Die primêre fokus van die studie is op die vaardigheidsvlak van die annoteerders, programmatuuromgewings wat gebruik kan word om vinniger beter data te lewer, en of dit voordeliger is om meer data te annoteer of om die kwaliteit van die data te verhoog. Ons omskryf elk van hierdie fokusareas en voer sistematiese eksperimente uit om die doeltreffendheid van elkeen te bepaal.

Ons ondersoek eerstens die invloed van respondente se vaardigheidsvlakke op data-annotasie deur geannoteerde data van deskundiges, dilettante en leke met mekaar te vergelyk. Uit die bevindinge is dit duidelik dat die deskundiges in beide take veel beter vaar as die nie-kundiges en dat dié verskil statisties beduidend is.

Vervolgens word die effek van die programmatuuromgewing wat vir annotasie gebruik word ondersoek om vas te stel wat die voordele verbonde aan pasmaakprogrammatuur versus domein-spesifieke en algemene programmatuur is. Die vergelyking toon dat dit in die geval van hierdie twee take voordelig is in terme van annotasietyd en kwaliteit om pasmaakprogrammatuur te gebruik eerder as domein-spesifieke of algemene programmatuur. In die konteks van linguistiese annotasie van data vir hulpbronskaars tale regverdig die bykomende tyd wat nodig is om pasmaakprogrammatuur te ontwikkel egter nie die besparing in annotasietyd nie.

Laastens word kerntegnologieë wat ontwikkel is met data van wisselende kwaliteit en kwantiteit met mekaar vergelyk om te bepaal wat die impak van meer data versus “skoner” data op die prestasie van sodanige tegnologieë is. Wanneer ŉ vergelyking getref word tussen stelsels wat afgerig is met minder hoëkwaliteitdata teenoor stelsels afgerig met meer laekwaliteitdata, vaar die stelsels met die laekwaliteitdata statisties beduidend beter. Derhalwe kom ons tot die gevolgtrekking dat dit meer voordelig is om op die kwantiteit eerder as die kwaliteit van die afrigtingsdata te fokus.

Na aanleiding van die resultate en analises van die eksperimente word aanbevelings gemaak met betrekking tot watter strategieë in die praktyk geïmplementeer behoort te word. Vir ŉ projek wat daarop gemik is om goudstandaarddata te ontwikkel, kan die beste resultate verkry word deur gebruik te maak van deskundiges wat data dubbelblind in pasmaakprogrammatuur annoteer (indien daarvoor voorsiening gemaak word in die begroting, of indien die ontwikkelingstyd die besparing in annotasietyd regverdig). Vir ŉ projek wat daarop gemik is om kerntegnologieë te ontwikkel, moet deskundiges of opgeleide dilettante gebruik word om data slegs een rondte in pasmaakprogrammatuur te annoteer (weereens, slegs as daarvoor voorsiening gemaak word in die begroting en skedule om sodanige programmatuur te ontwikkel).

Sleutelwoorde:

Afrikaans; Outomatiese spraakherkenning; Lemma-identifisering; Hulpbronskaars tale; Mensetaaltegnologie; Hulpbronontwikkeling.

Opsomming

Acknowledgements

I would like to express my appreciation and thanks to all the people who helped and supported me during my doctoral study. I am deeply appreciative of, would like to express my sincere gratitude and give particular mention to:

- Gerhard van Huyssteen (promoter, mentor and friend);
- Etienne Barnard (co-promoter);
- Martin Schlemmer;
- Roald Eiselen;
- Willem Basson;
- past and present colleagues at the Centre for Text Technology;
- my friends and family; and
- Research Unit: Languages and Literature in the South African context and the North-West University for financial assistance.

This thesis is dedicated to my wife, Lindi, and my daughter, Brigitte.

Table of contents

Table of figures

1 Chapter 1: Introduction

1.1 Contextualisation

Let us assume a hypothetical project where we want to develop two core technologies for a resource-scarce language: a lemmatiser and an automatic speech recognition system. During project planning we must ask several questions, for example: Who should we use to annotate the data? Do we need specialised software for the annotations? What quality control measures should we employ? Most projects involved in the development of human language technologies (HLTs) for resource-scarce languages face these questions and often do not have adequate experience or proof on which they can base their decisions.

Since the development of HLTs often depends on the availability of linguistic data, especially annotated corpora, the development of such resources is imperative for the HLT enablement of any language. Developing highly accurate, annotated data is, however, often a time-consuming and expensive process – even more so in the context of resource-scarce languages. The development of technologies for resource-scarce languages contributes to bridging the digital divide (i.e. the divide between the privileged and the marginalised in terms of access to technology, specifically computers and related applications) and ensure that speakers of resource-scarce languages are not excluded from using language technologies and the associated benefits of improved human-machine interaction.

Wagacha *et al.* (2006) define a resource-scarce language as “a language for which few digital resources exist; a language with limited financial, political, and legal resources; and a language with very few linguistics experts”. Given the limitations of available resources, finances and expertise, projects entailing HLT development for resource-scarce languages explore and implement various strategies through which the development of HLT resources can be expedited. These strategies include using non-experts instead of experts to annotate data, the development of software to fast-track and improve manual data annotation, using methods such as bootstrapping and unsupervised learning, and technology transfer between closely related languages. These strategies aim to speed up the manual annotation process, improve annotation accuracy, and/or reduce the workload of annotators (thus reducing the annotation time and associated cost). Although the above-mentioned strategies are implemented in various projects and have been proven to be beneficial for mainstream languages, their efficiency in creating resources for resource-scarce languages has not been conclusively established. This study investigates the efficiency of some of these strategies in the development of resources for resource-scarce languages, in order to provide recommendations for future projects facing decisions regarding which strategies they should implement.

1.2 Problem statement

Three main considerations in the process of data annotation are (1) the nature of the data, (2) the nature of the annotation task, and (3) factors related to the performance of the annotator and his/her environment. Each of these contributes to the process of data annotation in terms of annotation time, quality of the annotations, and cost.

The nature of data includes the language of the data and the modality of the data. The language determines which resources are available (such as existing corpora and software) and the availability of linguistic experts. For resource-scarce languages, the resources available are usually few or non-existent. The modality of the data might be text, audio, video, images, gestures or body posture, which has an influence on the complexity of the process of data annotation.

The nature of the annotation task can be influenced by the nature of the data, as well as by the complexity of the task. Text data, for example, can be annotated on an internal word level (grapheme-to-phoneme annotation, compound analysis, hyphenation, etc.), or on external sentence or paragraph level (such as part-of-speech tagging, terminology extraction, named-entity annotation). The complexity of the task has a direct influence on the nature of the task as well as on the choice of annotator (e.g. skill level and training) and the environment used for annotation (i.e. the software must be able to accommodate the nature of the task).

Factors related to the performance of the annotator include the skill level of the annotator, training of the annotator, time available to perform annotations (e.g. experts might be full-time employed elsewhere), professional fees, computer literacy, etc. Factors of his/her environment include such matters as user-friendly interfaces, features aimed at improving quality and time needed to perform annotations, compatibility with standards and formats, etc.

In this study, the main focus is on the latter, i.e. on factors related to the annotator and his/her environment. We focus on two of these factors, viz. the skill level of the annotators, and features aimed at improving time needed to perform annotations and quality. We also examine how best to use the annotator for the development of core technologies, namely whether to annotate more data, or to increase the quality.

One of the first tasks in any HLT-related project is finding suitable annotators. For mainstream languages this is usually not problematic since ample numbers of linguistic experts are available. For resource-scarce languages on the other hand (and in accordance with the definition “resource-scarce”), very few linguistic experts exist. This necessitates finding alternative annotators to perform the annotations. One approach is to use non-experts as annotators, and studies investigating the effectiveness of non-experts have found that non-experts are suitable for certain annotation tasks (e.g. Snow *et al.* (2008); for further discussion see 2.2). These studies are however mostly based on mainstream languages and are usually conducted via a crowdsourcing platform such as *Amazon’s Mechanical Turk*. For mainstream languages a suitable workforce of non-experts is usually available, and projects often use multiple non-experts to annotate data. From the multiple annotations of the same data, projects are able to extract annotated data of adequate quality (usually by means of voting (Mellebeek *et al.*, 2010)). For resource-scarce languages, a suitable workforce might not be available in a crowdsourcing environment. Given the limited number of linguistic experts available, it is still prudent to investigate whether, similar to the idea of crowdsourcing, a crowd-like group (i.e. untrained, recruited respondents) can be used for annotation of data for resource-scarce languages.

According to the definition provided in section 1.1, a resource-scarce language is a language for which few digital resources exist. This implies that resources need to be created. For the development of these digital resources, necessary tools are required to deliver high quality annotated data in the shortest possible time. One way in which to fast-track the development of these resources is by using software that is readily available. However, although software and systems can help users to perform certain tasks, these packages are either not created with the purpose of annotation in mind (in the case of generic off-the-shelf software), lack some functionality required by the task (in the case of generic annotation software), or are created for a very specific task (in the case of available custom graphical user interfaces (GUI’s)). These different software environments each have pros and cons, but according to studies conducted on annotation projects using tailor-made software, it seems as if it might be beneficial both in terms of saving annotation time and in increasing annotation accuracy to use tailor-made software (Bertran *et al.*, 2008; Eryigit, 2007; Maeda *et al.*, 2006). One crucial aspect not discussed in detail by these studies is the additional time and funds needed for development of tailor-made software, and whether the additional development time can be justified by reducing the annotation time. Projects that face the decision of either developing tailor-made software or using existing general-purpose or domain-specific software, need to be aware of how much reduction in annotation time they can expect in order to judge whether it will be beneficial to develop tailor-made software, given the limited resources of resource-scarce languages.

A final question to consider is whether annotators should be used to improve the quality or increase the quantity of annotations. In most annotation projects of resource-scarce languages, the goal is to develop core technologies with the annotated data by using the data as training data for a machine learner. Because of limited financial resources, projects often have to decide whether quality control needs to be performed, or if they should rather annotate more data. It is commonly accepted (Aduriz *et al.*, 2003; Bada *et al.*, 2012; Dang *et al.*, 2002; Zaghouani *et al.*, 2010) that higher quality annotated data will result in a more accurate system, and that more data will result in a more accurate system. What is not apparent, however, is which of these two commonly accepted maxims should be followed when a project's finances only allow for one. Projects that decide instead to improve the quality of annotations often use methods such as double-blind annotation, where multiple annotators are used to annotate the same data in order to detect and correct discrepancies. What is often not clear is how much impact errors have on the performance of the system. Also, if the data is only single-based annotated (i.e. if the annotators annotate different sets of data), double the quantity of data can be annotated compared to the use of double-blind annotation. Although the single-based annotated data will contain some degree of errors, it is not clear if the benefit of using more data, containing errors, will outweigh the benefit of using less, "cleaner" data.

In summary, the main problem around which this study is based is that the efficiency of the strategies used during the development of HLT resources for resource-scarce languages is not always clear. This study will investigate the efficiency of using non-experts instead of experts to annotate data, and using tailor-made software instead of domain-specific or general-purpose software for annotation. The effect of the quality and quantity of annotated data on machine learning systems will also be explored.

1.3 Research questions

In order to address the above-mentioned problems, the following main research question is formulated:

- Which strategies are the most efficient for developing resources for HLTs for resource-scarce languages?

Specific research questions relating to annotators, user interfaces, and data quality vs. data quantity are posed:

1. Can comparable results (in terms of quality of the annotations and time needed to perform the task) be obtained using experts and non-experts for the task of linguistic annotation of data for resource-scarce languages?
2. If comparable results can be obtained using non-experts, is it beneficial to use novice annotators instead of laymen?
3. Is it beneficial in terms of time and quality to use tailor-made software instead of domain-specific or general-purpose software?
4. If it is beneficial to use tailor-made software, can the additional development time be justified by the savings in annotation time?
5. Is it more beneficial to focus on the quality or the quantity of training data?

1.4 Aims

The main aim of this research is:

- To determine which strategies are the most efficient for developing resources for HLTs for resource-scarce languages.

The specific aims related to the above-mentioned questions are:

1. To compare the results obtained using experts and non-experts for the task of linguistic annotation of data for resource-scarce languages in order to establish whether non-experts are a suitable alternative for annotation;
2. If comparable results can be obtained using non-experts, to establish whether it is beneficial to use novice annotators instead of laymen;
3. To establish the benefits in terms of time and quality when using tailor-made software instead of domain-specific or general-purpose software;
4. To establish whether additional development time of tailor-made software can be justified by the savings in annotation time; and

5. To establish whether it is more beneficial to focus on the quality or on the quantity of training data.

Secondary to the main aim of the study, is to make recommendations regarding which of these strategies should be implemented in practice.

1.5 Methodology

1.5.1 Scope

For all experiments, Afrikaans is used as an example of a resource-scarce language with a conjunctive orthography and productive affixation. Afrikaans is one of the eleven official languages of South Africa and is estimated to have 6.85 million native speakers (Statistics South Africa, 2013). This differs considerably from the number of native speakers of mainstream languages such as Spanish with 406 million native speakers, English with 335 million, German with 83.8 million, French with 68.5 million, and Dutch with 22.9 million (Lewis, 2009). According to Grover *et al.* (2010), Afrikaans has the most prominent technological profile of all South African languages. Nonetheless, all South African languages have basic core resources available, i.e. unannotated monolingual text corpora, lexica, speech corpora, etc. Even though Afrikaans is used as the exemplary language in this study, none of its more advanced language resources (such as a compound analyser or part of speech tagger) are used in any of the experiments.

Afrikaans was chosen as the resource-scarce language for this study for several reasons. For the experiments described in Chapters 2 and 3, ninety native speakers of a resource-scarce language, who were undergraduate students studying for a bachelor's degree with the specific language included in his/her curriculum were needed. At the North West University[1], the only South African resource-scarce language with a sufficient number of students was Afrikaans. In order to compare the annotations of Chapters 2 and 3, as well as training data for the experiments in Chapter 4, gold standard data for both tasks were needed. The gold standard data used in this study was developed in previous projects conducted by the Centre for Text Technology (CTexT)[2]. Also, for the task of orthographic transcription, the errors made by respondents (as discussed in Chapters 2 and 3) were to be manually annotated by the author, who is a native speaker of Afrikaans. However, even though the scope of this study is

[1] www.nwu.ac.za

[2] www.nwu.ac.za/ctext

restricted to Afrikaans, the results will not only be applicable to Afrikaans, but also to other resource-scarce languages[3].

The complexity of the tasks is restricted to intermediate linguistic tasks (see 2.2 for a description). This was done in order to compare the influence of specific dimensions in each chapter, i.e. the skill level of respondents in Chapter 2, different software environments in Chapter 3 and the effect of data quality vs. data quantity in Chapter 4. In Chapters 2 and 3, we investigate lemmatisation of text data and orthographic transcription of audio data. In Chapter 4, we develop two core technologies, viz. a lemmatiser (capable of identifying the lemma of inflected words (Groenewald, 2006)), and an automatic speech recognition system (software used for independent, computer-driven transcription of spoken language into readable text in real time (Stuckless, 1994)).

The focus in Chapter 2 is on one specific factor related to the annotator, namely different skill levels – and whether using a crowd-like group of untrained, recruited respondents (similar to the idea of crowdsourcing) is a suitable alternative to using experts for annotation of data for resource-scarce languages. However, we do not aim to provide a comprehensive evaluation or overview of crowdsourcing, but rather to make use of a crowdsourcing environment to investigate the matter at hand.

In Chapter 3, tailor-made software developed with specific features (aimed at the specific tasks) is described. These software environments and the specific features included are only exemplary of assistive technologies, and do not imply that these are the most suited features. The aim is to determine if the addition of task specific features is beneficial to the annotation task by increasing the quality or reducing the time needed to perform the annotations. Some of the features, for example automatic protocol flagging (see Annexure C.3 and Annexure C.5 for a description of the software environments and these features) are implementable for the majority of languages, but some features are dependent on the availability of specific resources. In both tailor-made software environments, features dependent on a spelling checker lexicon are included[4] and might not be available for other resource-scarce languages.

[3] The same methodology followed here could also be applied to mainstream languages (such as English or Spanish) to simulate resource-scarceness, or alternatively to languages without any resources (e.g. some of the San languages), which would be much more difficult to execute and evaluate.

[4] For other tasks, different resources might be needed, for example using frequency information when developing lexica, or by displaying the part of speech of a word when performing morphological analysis.

Nonetheless, we decided to include these features based on the following considerations:

1. According to the BLARK (Basic LAnguage Resource Kit) (Krauwer, 2003), monolingual corpora and, subsequently, lexica are considered as basic core language resources (LRs) needed for every language. Lemmatisers are considered to be more advanced LRs. Although the BLARK methodology of prioritising resource development is not followed by all languages, it is common practice to start resource development by collecting corpora, extracting lexica from the corpora and then enriching the data with annotations such as lemmatisation information.
2. Lexica and spelling checkers are available for a variety of languages and new languages are constantly being added to the available languages by vendors such as Microsoft[5] and GNU Aspell[6], research projects or even individuals.
3. If a project wants to include spelling checking features and does not have access to lexica, rudimentary lexica can be developed in parallel to the project by iteratively reviewing the annotated data and including the correctly spelled words in a lexicon. A rudimentary lexicon can also be developed by including the highest frequency words extracted from a corpus. Schmitt and McCarthy (1997) investigated the coverage of the most frequent words in English and found that in the Brown Corpus of Standard American English[7], totalling roughly one million words, the 2,000 most frequent words gives near to 80% coverage of the corpus.

One prerequisite for features included in the tailor-made software was that the intended core technologies to be trained with the annotated data (i.e. a lemmatiser and an ASR system) could not be included. Thus, methods such as bootstrapping or active learning, which are used to improve or reduce the data to be annotated, are explicitly excluded. Software that primarily focus on these methods are also excluded from the literature survey and discussions.

For the comparison of the data annotated by respondents of different skill levels (Chapter 2) and the comparison of data annotated in different software environments (Chapter 3), the data is compared in terms of time needed to complete the task, as well as the quality of the data. In order to determine if the development of tailor-made software can be justified by the benefit to the data annotation process (Chapter 3), only the development time is compared to the saving in annotation time. Other benefits, specifically the increase in quality, are ignored for purposes of our comparison.

[5] http://office.microsoft.com – 63 spelling checkers available

[6] http://aspell.net/ – 91 spelling checkers available

[7] http://icame.uib.no/brown/bcm.html

Ideally, one would conduct experiments on various resource-scarce languages, tasks and software environments, as well as on large datasets, but the scope of such an endeavour is vast and not achievable in this study. As such, we focus on one language, two tasks and seven software environments. As with any quantitative research, the number of observations per group is imperative for further statistical analysis. In Chapter 2 and Chapter 3, the focus is on the respondents and the associated quality of the annotations given different levels of expertise and different software environments. As such, for the sample size we decided on ten respondents for each of the novices and laymen groups (Chapter 2), and ten respondents for each software environment (Chapter 3). This ensured that ten observations per group could be made, instead of using, for example, five respondents to each annotate double the quantity of data, thereby reducing the relevant data points. The systematic description of the experiments and results provides a baseline that is applicable to future experiments with other resource-scarce languages.

1.5.2 Method

In order to determine whether non-experts (i.e. untrained, sourced respondents, similar to the idea of crowdsourcing) can be used for annotation of resource-scarce language data, we investigate the effect of respondents' skill levels on data annotation. Variables which could influence the results of the experiments (viz. hardware, training, presentation of data, and software) are kept constant in order to ensure a controlled experiment. To further ensure that a particular respondents' learning curve of a task does not influence the results, the datasets are kept relatively small. By limiting the datasets to a size that could be completed in approximately one hour, it is assumed that the respondents will not gain enough experience to significantly improve on annotation speed or accuracy. Tasks completed via crowdsourcing are also mostly performed by a large number of respondents, each completing only a small part of the overall dataset, and by keeping the datasets relatively small, our experiments follow the crowdsourcing approach more stringently. The two tasks are each completed by three distinct groups of respondents (42 respondents in total). The resulting data is evaluated in terms of time needed to perform the task, and quality of the data. The quality of the data is measured by comparing the data annotated by the respondents to gold standard data as described in 2.4.2, and manually annotating and classifying all errors present in the respondents' transcriptions into separate categories (see 2.4.5.2.1).

To determine the benefits of using tailor-made software instead of general-purpose or domain-specific software, we investigate the effect of software environments on data annotation. The two tasks are completed in seven different software environments: four for the task of lemmatisation and three for the task of orthographic transcription of audio data. The hardware, skill level of the respondents (seventy respondents in total), training and presentation of data are kept constant. As in Chapter 2, the resulting data is also evaluated in terms of time needed to perform the task, and quality of the data.

To compare systems trained with data of the same quantity but with varying levels of quality, and also to compare systems trained with gold standard data (see 4.4.2 for a description) to systems trained with lower quality but double the quantity of data, datasets for the two tasks are developed for use as training data. To simulate real world errors, the quality of the annotations reported in Chapters 2 and 3 is used as the means of describing levels of errors that are generated in the different datasets. Ten increments of data, ranging from 10% to 100%, are randomly extracted to simulate the increase of data quantity. Tenfold cross-validation is performed, resulting in 500 distinct experiments for each of the two tasks. The resulting systems are evaluated using standard evaluation metrics for each task.

1.6 Deployment

In the subsequent three chapters, factors that contribute to the process of data annotation (i.e. the annotator, the user interface that the annotator uses, and quality vs. quantity of the annotated data) are discussed. Specific hypotheses are proposed in each chapter. Each chapter provides a brief literature review comprising a general survey of the relevant topic for the chapter, as well as case studies. Given the fast development in NLP, it is almost impossible to give a comprehensive overview of state of the art. Although we tried to be all-inclusive, some of the latest findings might not be included. Each chapter then outlines the different experimental setups which were followed in this study, as well as relevant evaluation criteria.

Chapter 2 explains some of the problems regarding the lack of a suitable non-expert workforce for resource-scarce languages. Additionally, the chapter describes the experiments conducted and results achieved to determine whether untrained, sourced respondents can be used for annotation of linguistic data for resource-scarce languages.

Chapter 3 describes some differences between general-purpose software, domain-specific software and tailor-made software. In the second part of this chapter, experiments in seven software environments are described, and the results from the different software environments are discussed in order to determine whether it is beneficial to use tailor-made software instead of general-purpose or domain-specific software.

Chapter 4 investigates the effect of data quality vs. quantity by comparing systems trained on varying quality and quantity of data. The aim of this chapter is to establish whether it is more beneficial to focus on the quality or the quantity of training data.

Chapter 5 provides a concluding summary and offers some recommendations regarding which of the methods described in Chapters 2, 3 and 4 should be implemented in practice. Finally, considerations for future work are described.

2 Chapter 2: The effect of respondents' skill levels in data annotation

2.1 Introduction

The aims of this chapter are to establish if non-experts are suitable for the task of annotating linguistic data for resource-scarce languages, and if it is beneficial to use novices instead of laymen as non-experts. The following section provides an overview of some completed projects using non-experts to annotate data, as well as the problems regarding the lack of a suitable non-expert workforce for resource-scarce languages. Section 2.4 describes the experimental setup and section 2.5 provides the results, analysis and interpretation that allow us to make recommendations in section 2.6.

2.2 Literature survey

Since the development of HLTs often depends on the availability of annotated linguistic data, the development of such resources is imperative for the HLT enablement of any language, and even more so for resource-scarce languages. As we have indicated in Chapter 1, the development of such annotated, digital resources is an expensive and time-consuming endeavour, and alternative methods are often sought to efficiently deliver high-quality annotated data.

One way in which to fast-track the development of these resources is by using non-experts for linguistic annotation. Non-experts are generally obtained by using the web as workforce – a method generally referred to as crowdsourcing (i.e. “the act of taking a task traditionally performed by a designated agent (such as an employee or a contractor) and outsourcing it by making an open call to an undefined but large group of people” (Howe, 2008)). People are recruited to complete tasks as non-experts with crowdsourcing software such as *Mechanical Turk*[8] (*MTurk*), *CrowdFlower*[9], *BizReef*[10], *Elance*[11], *Freelancer*[12], *SamaSource*[13], etc. Data collected via crowdsourcing is categorised as human intelligence tasks (HITs), indicating tasks that are simple for a human to perform, but difficult for computers (Alonso *et al.*, 2008).

[8] www.mturk.com

[9] www.crowdflower.com

[10] www.bizreef.com

[11] www.elance.com

[12] www.freelancer.com

[13] www.samasource.org

Tasks that have been completed successfully via crowdsourcing include, inter alia:

- named-entity annotation (Finin *et al.*, 2010; Higgins *et al.*, 2010; Lawson *et al.*, 2010; Yetisgen-Yildiz *et al.*, 2010);
- classification of (Spanish) consumer comments (Mellebeek *et al.*, 2010);
- word-sense disambiguation (Akkaya *et al.*, 2010; Hong & Baker, 2011; Snow *et al.*, 2008) and creation of word-sense definitions (Rumshisky, 2011);
- Urdu-to-English translation (Zaidan & Callison-Burch, 2011), correction of translation lexicons (Irvine & Klementiev, 2010), ranking of machine translation results (Callison-Burch, 2009) and word alignment for machine translation (Gao & Vogel, 2010);
- rating of similarity between phrasal verbs; segmentation of audio speech streams; judgment studies of fine-grained probabilistic grammatical knowledge; confirming corpus trends (Munro *et al.*, 2010);
- classifying sentiment in political blog snippets (Hsueh *et al.*, 2009);
- rating newspaper headlines for emotions; rating of similarity between word pairs; recognising textual entailment; event temporal ordering (Snow *et al.*, 2008);
- rating of computer-generated reading comprehension questions about Wikipedia articles (Heilman & Smith, 2010);
- extraction of prepositional phrases and their potential attachments (Jha *et al.*, 2010); and
- cloze tasks (one or several words are removed from a sentence and a student is asked to fill in the missing content) (Munro *et al.*, 2010; Skory & Eskenazi, 2010).

Orthographic transcriptions of audio data and collection of speech data are also often performed via crowdsourcing. The data which is transcribed ranges from easy transcription and correction tasks, to full manual annotation of audio. Some examples of transcription and collection tasks include:

- route instructions for robots (Marge *et al.*, 2010a);
- correction of automatic captioning (subtitles) (Wald, 2011);
- bus information system data (Parent & Eskenazi, 2010);
- conversational telephone speech (Novotney & Callison-Burch, 2010);

- meeting speech (Marge *et al.*, 2010b);
- young child's early speech (Roy *et al.*, 2010);
- recordings from news websites (Gelas *et al.*, 2011);
- Mexican Spanish broadcast news corpora (Audhkhasi *et al.*, 2011b);
- Mexican Spanish audio (Audhkhasi *et al.*, 2011a);
- academic lecture speech (Lee & Glass, 2011);
- collection of speech data containing spoken addresses (McGraw *et al.*, 2010); and
- collection of responses to an assessment of English proficiency for non-native speakers (Evanini *et al.*, 2010).

Completed HIT studies have shown that non-experts can be used to annotate data that is comparable in terms of quality to annotation performed by experts. Of the 38 studies mentioned above, most (with the exception of three (Finin *et al.*, 2010; Irvine & Klementiev, 2010; Wald, 2011) that did not explicitly report comparisons of quality) reported that the annotated data collected from non-experts or systems trained with the non-expert data, was useful, in high agreement, comparable, or of similar quality to annotated data collected from experts. One noticeable aspect of most of these studies was that a single expert is in most cases more reliable than a non-expert, but by using non-expert data, usually combined with some form of voting or bias correction, the quality of the combined non-expert data approaches (or equals) the performance of experts. Mellebeek *et al.* (2010) even reported that in their study of classifying Spanish consumer comments, the non-experts outperformed experts.

Snow *et al.* (2008) conducted experiments on five natural language processing tasks, i.e. affect recognition, word similarity, recognising textual entailment, event temporal ordering, and word sense disambiguation. They reached the conclusion that only a small number of non-expert annotations (four) per item were necessary to equal the performance of an expert annotator. Callison-Burch (2009) conducted a comparison between experts and non-experts on the evaluation of translation quality and concluded that it is possible to achieve equivalent quality using non-experts, by combining the data of five non-experts. Similar results were achieved by Heilman and Smith (2010), who used crowdsourcing to rate computer-generated reading comprehension questions about Wikipedia articles and found that combined data of three to seven non-experts rivalled the quality of experts.

Although studies show that non-experts can be used to achieve results similar to those achieved by experts, various factors (often not discussed at length in the literature) could influence the success of using non-experts for annotation of linguistic data for resource-scarce languages. The following factors should be kept in mind:

- complexity of tasks;
- language(s) of the tasks; and
- skill level of the annotator.

These three factors have an influence on the annotator, and how successful he/she is in performing the task. The focus of this chapter is on these three factors, and the influence that these factors have on the annotators' ability.

The **complexity of tasks** performed via crowdsourcing is generally low as tasks require the worker to make one or more choices from a small range of possible answers (i.e. multi-choice answers). They are typically represented as radio buttons, check boxes or sliders (Eickhoff & de Vries, 2011). For purposes of this study, three levels of complexity are proposed.

1. **Basic** linguistic tasks are tasks that an average native speaker of the language is capable of performing if brief instructions are provided and which require no specialised linguistic knowledge – for example, rating consumer comments as being either positive, negative or neutral (Mellebeek *et al.*, 2010), rating newspaper headlines for emotions, rating of similarity between word pairs (Snow *et al.*, 2008), rating computer-generated questions on a five point scale (Heilman & Smith, 2010), etc. Transcription of audio data could be included in this level if a speaker is only required to transcribe what he/she hears, and if the task does not include any additional stipulations such as indicating mispronounced words, indicating certain types of noise, etc.
2. **Intermediate** linguistic tasks are presented as tasks that an average native speaker of a language will need limited training in or possesses specialised knowledge of, as he/she needs to use pre-existing knowledge to interpret and perform a specific task. At least a clear, more comprehensive description, protocol or training must be provided. The tasks investigated in this chapter (viz. lemmatisation and transcription of audio data) are categorised as intermediate linguistic tasks. For the task of lemmatisation, the protocol stipulates that all inflected forms must be normalised to a lemma, but derivations should be left as they originally appear. Thus,

the respondent needs to be able to interpret these stipulations and use his/her existing knowledge of inflectional and derivational suffixes to perform the task. The protocol for the task of transcription of audio data also contains some stipulations that require the respondents to use their existing linguistic knowledge in order to complete the task. For example, abbreviations should be written in capital letters with spaces between the letters, but acronyms should be written with capitals, but without spaces between the letters. Thus, the respondent needs to be able to use his/her existing knowledge of the difference between abbreviations and acronyms to perform the task. Aspects like inflection vs. derivation, or abbreviations vs. acronyms are deemed delineated enough to be explained in a more comprehensive protocol, for native speakers to understand.

3. **Advanced** linguistic tasks require more linguistic knowledge than an average native speaker possesses, and the speaker needs specialised training or experience in similar tasks in order to perform these tasks. For example, an average speaker might be able to perform part-of-speech tagging on a basic level, e.g. to distinguish between a noun or a verb, but will probably not be able to perform POS tagging with a fine-grained tagset that includes categories such as *non-third person singular present verb* without extensive training. Other advanced linguistic tasks include morphological analysis, phonetic transcription, chunking, etc.

Another factor could be the **language(s) of the tasks**, which usually involve mainstream languages such as English; only a few studies have been conducted using resource-scarce languages. Novotney and Callison-Burch (2010) used crowdsourcing to collect data for automatic speech recognition (ASR) with *Mechanical Turk*. For English they collected transcriptions of twenty hours of speech, transcribed three times. These transcriptions were performed by 1089 *Turkers* who completed ten hours of transcriptions per day. They also experimented with Korean, Hindi and Tamil. Transcription of Korean progressed very slowly; two workers completed 80% of the work only after they received additional payment. They had a test set for Korean and found that the average disagreement with the reference transcription was 17%. They only managed to complete three hours of transcriptions in five weeks. For Hindi and Tamil only one hour of transcription was completed in eight days. They also did not have any expert transcription to compare the non-expert transcriptions to and could not provide any results on the quality of the non-expert transcriptions.

Gelas *et al.* (2011) acquired transcriptions for Swahili and Amharic and found that it is possible to acquire quality transcriptions from crowdsourcing, although the completion time is much slower than similar projects conducted in English. The transcriptions of Swahili were completed in twelve days, but the transcriptions of Amharic only reached 54% completion after 73 days. The word error rate (WER) achieved on the transcriptions was 16% for Amharic and 27.7% for Swahili, and on the ASR systems 39.6% for Amharic and 38.5% for Swahili. This is similar to the WER achieved on ASR systems trained using reference transcriptions: 40.1% for Amharic and 38% for Swahili. This indicates that although the quality of the transcriptions is adequate, it is still challenging to complete tasks involving resource-scarce languages because there is not an adequate workforce available.

The lack of studies involving resource-scarce languages raises the question of why crowdsourcing is not used as extensively for HLT annotation as for mainstream languages. The most prominent factor is the demographics of users of crowdsourcing software. (Ross *et al.*, 2010) conducted a survey of workers on *Amazon's Mechanical Turk*, referred to as *Turkers*, and found that *Turkers* are mainly based in India (46%) and the USA (39%). Ipeirotis (2010) conducted a similar survey and found that of one thousand respondents, only one was based in South Africa and only about 1% were from Africa. Munro and Tily (2011) extended their survey and also asked respondents for information about which languages they spoke apart from English. Data from about two thousand respondents showed a total of one hundred languages. From these two thousand respondents, only two could speak Afrikaans, with one respondent originating from South Africa and the other from China. This pattern extends to other resource-scarce languages as well, and shows a low number of speakers, e.g. Albanian (1), Bulgarian (2), Creole (1), Czech (1), and Swahili (1). Although the number of native speakers of mainstream languages (e.g. English with 335 million native speakers (Lewis, 2009)) differ considerably from speakers of resource-scarce languages (e.g. Afrikaans estimated at 6.85 million native speakers according the 2011 census of South Africa (Statistics South Africa, 2013)), the number of *Turkers* who speak resource-scarce languages is exceptionally low.

One factor that contributes to the low number of resource-scarce language *Turkers* is the payment structure. International *Turkers* (excluding *Turkers* from India) can only be paid with an *Amazon.com* gift certificate. Other complications also deter international *Turkers*, for example the South African post office was "blacklisted" at one point, and all shipments to South Africa could only be done with a private courier, resulting in very high cost[14]. The implication is that performing tasks via crowdsourcing is not financially beneficial to speakers of resource-scarce languages who reside outside the USA or India, and thus the pool of potential workers is reduced.

[14] www.timeslive.co.za/News/Article.aspx?id=786533

Another factor to consider is access to internet. It is estimated that in 2012 (quarter two) only 15.6% of the population of Africa had access to the internet. South Africa is only slightly higher with 17.4%. An estimated 37.7% of the population of the rest of the world has access to the internet[15]. These statistics include access via fixed and wireless broadband as well as mobile data. In Africa the foremost access (estimated between 60% and 99%) to internet is via mobile data and the users have very limited access to computers (estimated at 2%), making mobile phones the dominant device for internet access. Although Africa has a high smart phone adaptation (estimated at 17% to 19% of total mobile phones), the implication is that only about 2% of the population of Africa has access to traditional crowdsourcing sites via suitable devices, further reducing the pool of potential workers.

The issues with payment combined with limited access result in an unsuitable workforce for crowdsourcing of tasks for resource-scarce languages. Even though we therefore cannot use traditional crowdsourcing on the web to determine the influence of the skill level of respondents on linguistic annotation of data for resource-scarce languages, it is still prudent to investigate if a crowd can be used, even though such a crowd has to be sourced for the sake of our experiments. As few linguistic experts for resource-scarce language are available, an alternative workforce that is readily available could prove advantageous to the development of resources for resource-scarce languages.

Some studies comparing non-expert respondents with expert respondents, but not making use of crowdsourcing software, have also been done. These studies utilise domain-specific and tailor-made software for the task and are relevant as the software remains constant for the individual experiments. Geertzen *et al.* (2008) compared naïve respondents with experts on the task of dialogue act tagging. For naïve respondents they employed six undergraduate students with four hours of lecturing and a few small exercises; for expert respondents they employed two PhD students who had had experience with the annotation scheme for more than two years. They concluded that differences in both inter-annotator agreement and tagging accuracy were considerable. Dandapat *et al.* (2009) followed a similar approach in using respondents with different levels of training in a case study involving POS annotation for Bangla and Hindi. Two respondents were trained intensively in-house with various phases of annotation and feedback, while the other two respondents were only provided with the data, annotation tools, guidelines and task description. As expected, the results showed that the respondents with more training were faster and more accurate than the respondents who received no training. They

[15] www.internetworldstats.com/stats1.htm

concluded that “reliable linguistic annotation requires not only expert respondents, but also a great deal of supervision” (Dandapat *et al.*, 2009).

Although crowdsourcing rationally does not seem to be a viable option for the annotation of Afrikaans data because there are often not sufficient respondents available for resource-scarce languages, we still decided to test this assumption practically. For this experiment we posted two jobs, one for the task of lemmatisation of Afrikaans and one for the task of orthographic transcription of Afrikaans audio data on the crowdsourcing platform, *CrowdFlower*. *CrowdFlower* is a general-purpose crowdsourcing application that allows customers to upload their own tasks to be carried out by users of various labour channels such as *Amazon Mechanical Turk*, *TrialPay*, and *Samasource*, thereby increasing the available workforce.

Surprisingly, the jobs were accepted within a matter of minutes, but the completed data contained only garbage. The data contained copies of the instructions, nonsense text, random quotes from internet searches, empty responses, etc. With the exception of one, all respondents originated from India. These invalid responses correlate with experiences of other researchers attempting to collect data via crowdsourcing. Various methods for detecting cheating have been proposed, such as the inclusion of a gold standard, only accepting workers who have a certain rating by job creators on previous tasks, automatic detection and exclusion of malicious workers by filtering on geographic location, denying payment to such workers, limiting the country of origin of respondents, etc.

After this first round, we posted the tasks again and limited the country of origin to South Africa. After thirty days, no task was successfully completed. This indicated that no suitable workforce was available for the completion of lemmatisation or orthographic transcriptions for Afrikaans.

Thus, we cannot accurately and with certainty determine if non-experts can be used for linguistic annotation and transcription of Afrikaans via crowdsourcing. Given the limited linguistic experts available, it is still prudent to investigate (similar to the idea of crowdsourcing) whether a **crowd-like group of untrained, recruited respondents** can be used for annotation of data for resource-scarce languages. The result will not only be applicable to Afrikaans, but to other resource-scarce languages as well. Untrained non-experts were sourced in order to investigate the suitability of non-experts for annotation.

Because a workforce is not available and needs to be sourced, this chapter also investigates another factor which could influence the quality of annotations, namely the **skill levels of the non-experts**. Skill levels of respondents can be influenced by their level of education and previous experience, as well as by training in the specific task. Thus, we investigate if it might be beneficial to source respondents who already have some knowledge of linguistics. Our assumption is that respondents who have constant exposure to linguistics might perform better on linguistic annotation tasks.

In summary, three factors which have an influence on the annotator and how successfully he/she is able to perform the task, are applicable to this chapter, viz. the complexity of the task, the language of the task and the skill level of the annotator. The influence of these factors on the annotators' ability will be investigated by using lemmatisation of text data and orthographic transcription of audio data as an intermediate linguistic task; a resource-scarce language, Afrikaans, as the language of the tasks; and an expert and non-experts of two different skill levels to perform the tasks.

2.3 Research questions

Proof exists that similar results can be achieved by using non-experts instead of experts (Heilman & Smith, 2010; Mellebeek *et al.*, 2010), but the tasks are often simple (Eickhoff & de Vries, 2011; Snow *et al.*, 2008) and the experiments are conducted on mainstream languages such as English (Lee & Glass, 2011; Munro *et al.*, 2010). On more complex tasks and on resource-scarce languages, results in the literature are not conclusive (e.g. Geertzen *et al.*, 2008; Novotney & Callison-Burch, 2010).

In order to investigate the viability of using untrained non-experts (i.e. a crowd, instead of experts) for annotation of data for resource-scarce languages and to establish if a difference exists between novices and laymen, this chapter aims to answer the following questions:

1. Can comparable results (in terms of quality of the annotations and time needed to perform the task) be obtained using experts and non-experts for the task of linguistic annotation of data for resource-scarce languages?
2. If comparable results can be obtained using non-experts, is it beneficial to use novice annotators instead of laymen?

2.4 Experimental setup

2.4.1 Description of tasks

Respondents had to follow specific protocols for both tasks, viz. **lemmatisation of text data** (Task A) and **orthographic transcription of audio data** (Task B). These protocols were developed in separate projects and simplified and customised for our experiments. Detailed descriptions of the tasks as well as the protocols used in the experiments are provided in Annexure A and Annexure B.

2.4.2 Data

Task A: Lemmatisation of 1,000 words

The 1,000 word text used in this task was extracted from a 50,000 word corpus compiled in a project funded by the government of South Africa through its National Centre for Human Language Technology (NCHLT)[16]. The corpus was edited to correct spelling errors, tokenisation errors, etc. The randomly extracted text comprised running text and included 35 sentences consisting of ten words each, fifteen sentences consisting of twenty words each, and fourteen sentences consisting of 25 words each. The data contained 865 words to be left unchanged (i.e. the words already appeared in the base form) and 135 words that needed to be lemmatised. The gold standard data used for the comparison with the data annotated by the respondents was created by performing additional quality control on this 1,000 word text. Each of the 21 respondents annotated the same 1,000 word text.

Task B: Orthographic transcriptions of six minutes of audio

The audio data used for the task of orthographic transcriptions consisted of a collection of various news bulletins from an Afrikaans radio station, *Radio Sonder Grense* (*RSG*)[17]. The data was transcribed by seven transcribers over a period of 24 months according to the protocol described in Annexure B. Various levels of quality control were performed in order to produce (largely) error-free transcriptions. From these news bulletins, 48 sentence level utterances were randomly extracted. The total duration of the extracted utterances was six minutes. As with the data used in Task A, additional quality control was performed on these utterances to produce gold standard data and each respondent transcribed the same six minutes of audio data.

[16] www.rma.nwu.ac.za

[17] www.rsg.co.za

2.4.3 Software environment

Both tasks were performed in *CrowdFlower* by all the recruited respondents[18]; see Annexure C.1 for a description of the software. For both tasks, certain variables that could influence the results of the experiments were kept constant in order to ensure a controlled experiment:

1. **Hardware.** All experiments were performed on desktop PCs with identical specifications. The experiments were conducted in a student computer laboratory at North West University. For Task B, all respondents used identical headphones to ensure similar noise levels and clarity of the audio data.
2. **Presentation of data**. For Task A, the words to be lemmatised were presented as a tokenised list. The tokens were provided with an empty text box directly underneath the token where the respondent had to provide the lemma. For Task B, the recording was divided on sentence level and each sentence was provided separately and in sequence to the respondent. He/she had to provide the transcription for each sentence.

2.4.4 Respondents

For purposes of this experiment we sourced three groups of respondents with different levels of expertise. These three groups are defined as follows:

- **Experts** are characterised as people who have extensive knowledge of the field (e.g. morphology or phonetics), or who have already conducted similar tasks or participated in a similar project. In this experiment, we used two experts[19], one expert for each of the tasks.
- **Novices** are regarded as people who have frequent exposure to some form of language studies. They are assumed to have some intuitive understanding of linguistics. A criterion for inclusion in this group is that the respondent must be an undergraduate student studying for a bachelor's degree with the subject Afrikaans included in his/her curriculum. A further criterion is that the respondent must be a native speaker of Afrikaans. We used ten novices in each task.

[18] For recruitment of novices and laymen, we contacted undergraduate students studying for a bachelor's degree with the subject Afrikaans included in their curriculum, advertised the experiment on campus, and paid recruiters who supplied us with successful referrals. The respondents received a small honorarium for the successful completion of a task.

[19] Both experts were independent consultants involved in previous Afrikaans projects conducted by CTexT, and were not involved in any post hoc analysis of the data.

- **Laymen** are seen as people who are native speakers of Afrikaans, but who do not have any exposure to language studies. It is assumed that laymen might have lower performance in terms of quality of annotated data when compared with the quality of annotations performed by a novice. A criterion for inclusion in this group is that the respondent has never studied Afrikaans at tertiary level. Ten laymen were used in each of the tasks.

2.4.4.1 Training

None of the respondents received any training for the tasks. They were all provided with a protocol and instructed to work through the protocol at their own pace. Once they were comfortable with all instructions in the protocol, they could start the task.

2.4.5 Evaluation criteria

In this chapter we use two sets of evaluation criteria, one for the task of lemmatisation and a separate set for the task of orthographic transcription. For each task, the data from the expert, novices and laymen are compared to gold standard data. See 2.4.2 for a description of the gold standards.

2.4.5.1 Task A: Lemmatisation of Afrikaans text data

To compare the performance of the different groups of respondents, evaluations were performed on time needed to complete the task, the overall performance, capitalisation errors made, spelling errors made and no response provided. The **time** taken to complete the task was measured in seconds, from when each respondent started the task until the last word was completed by the respondent.

For purposes of evaluating the **overall performance** of the respondents, we calculate the accuracy by dividing the total number of words correctly lemmatised and words correctly left unchanged by the total words in the task.

$$Accuracy = \frac{Correctly\ lemmatised + Correctly\ left\ unchanged}{Total\ words}$$

We also use three additional standard evaluation metrics, viz. precision, recall and *f*-score, calculated on the words to be **lemmatised** (i.e. words that appear in an inflected form). These scores were calculated to provide a more informative indication of performance, since a high accuracy can be achieved by simply providing the original word. The data consisted of 865 words to be left unchanged and 135 words to be lemmatised (i.e. a respondent could achieve an accuracy of 86.5% by simply returning the same words provided to him/her).

Precision measures how accurately words to be lemmatised are lemmatised and is calculated by dividing correctly lemmatised words by total words lemmatised.

$$Precision = \frac{Correctly\ lemmatised}{Total\ lemmatised}$$

Recall is used to calculate how many words to be lemmatised are correctly lemmatised and is calculated by dividing the number of correctly lemmatised words by the total number of words to be lemmatised.

$$Recall = \frac{Correctly\ lemmatised}{Total\ to\ be\ lemmatised}$$

The ***f*-score**, which can be seen as a harmonic mean, is calculated as:

$$F\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Accuracy, precision, recall and *f*-scores are presented as percentages in this study.

The protocol stipulated that words at the beginning of a sentence, as well as certain named-entities such as names of departments, should be written in lower-case letters (e.g. "*department van behuising*" (*department of housing*) instead of "*Departement van Behuising*" (*Department of Housing*)). To compare the **capitalisation errors** made, the words that were incorrectly written with a capital letter, but would be correct if converted to a lower-case letter, were counted. The **spelling errors** made by respondents in the different software environments were calculated by comparing the data of each respondent to a spelling checker lexicon. An instance where no lemma was provided was counted as **no response**.

2.4.5.2 Task B: Orthographic transcription of Afrikaans audio data

As with Task A, time was measured in seconds, from when the respondent started the task until he/she was finished. For our experiments, the errors present in the respondents' transcriptions were manually annotated and classified into separate categories as described in the following section. Some of the errors, for example capitalisation and punctuation errors, are not applicable to ASR systems as the text is usually converted to lower case and punctuation is removed before training an ASR system. Nonetheless, the aim of Chapter 2 and Chapter 3 is to compare the overall quality of the orthographic transcriptions and not the impact of relevant errors on ASR systems[20]. Thus, all errors present in the data were manually annotated and taken into account for the comparisons.

2.4.5.2.1 Classification of errors

Errors in orthographic transcriptions can be caused by three factors:

1. **Inaccurate perception of events**. This includes mishearing of the audio recording (for example, the respondent transcribes "*the*" instead of "*a*") and confusables (for example, the respondent transcribes "*eye*" instead of "*I*"), as well as where respondents transcribe extra words, omitted words that were uttered, and switched the sequence in which the words were uttered.
2. **Inadequate skill level**. This is applicable to the respondents' level of expertise, especially in relation to their language skills, i.e. deviation from rules pertaining to the language and the standard written variant of the language. Errors in this category include incorrect use of capitalisation (for example if a proper name is written with a lower-case letter instead of an upper-case letter), incorrect usage of punctuation marks (such as the placement of commas and hyphens according to spelling conventions), non-words (e.g. *tabel* ->*table*), run-ons (e.g. *heruns* ->*he runs*), and splits (e.g. *fire man* ->*fireman*).
3. **Incorrect interpretation or implementation of the protocol**. This includes deviation from specific stipulations in the protocol, such as inclusion of invalid punctuation marks, white space usage, terminators, using digits instead of writing out numbers, incorrect writing of abbreviations or acronyms, or starting sentences with uppercase letters. The stipulations in the protocol are, in some cases, in contrast to the standard rules pertaining to the language. For example, the letters of an abbreviation were to be written next to one another, separated with white space, and in uppercase letters, even though this deviated from conventional spelling.

The errors found in orthographic transcriptions can be classified in one of three categories according to the factors listed above: transcription errors, language errors and protocol errors. For purposes of classifying errors in these experiments, eighteen different types of errors are grouped into these three categories. Results are provided for the three broad categories, followed by detailed breakdowns of each category into the relevant types in order to facilitate a clear understanding of the exact nature of

[20] In Chapter 4, respondents' errors that are not applicable to the actual training of the systems, such as capital letters, noise markers and punctuation are removed before training the systems. See 0 for a detailed description of the errors that were included in the training data.

errors made by the different groups of respondents. Table 1 provides an overview of the eighteen errors in their relevant categories as well as a brief description and an example.

	Description	English example (correct version provided first)	Afrikaans example (correct version provided first)
Transcription errors			
1. Insertions	A word that was not in the audio was transcribed.	*so we took our way toward the palace* *so we* ***we*** *took our way toward the palace*	*toe vat ons die pad na die kasteel toe* *toe vat ons* ***ons*** *die pad na die kasteel toe*
2. Deletions	A word that was in the audio was not transcribed.	*as to the first the answer is simple* *as the first the answer is simple*	*met belang tot die eerste is die antwoord eenvoudig* *met belang die eerste is die antwoord eenvoudig*
3. Substitutions	An incorrect word was transcribed.	*He is the man* *He is* ***a*** *man*	*hy is die man* *hy is* ***'n*** *man*
4. Transpositions	Two (or more) words in the audio were reordered in the transcription.	*He went on a long trip* *He went on* ***long a*** *trip*	*hy het op 'n lang reis vertrek* *hy het op* ***lang 'n*** *reis vertrek*
Language errors			
5. Spelling error	Deviations from spelling conventions for the language.	*there is no arbitrator except a legislature* *there is no* ***abritator*** *except a legislature*	*daar is geen arbiter buiten 'n wetgewende mag nie* *daar is geen* ***abriter*** *buiten 'n wetgewende mag nie*
6. Capitalisation	Following spelling conventions, proper nouns, titles of books, place-names, brand names, names of societies, commissions, etc. were to be written with an initial upper-case letter. Multi-word named entities were to be written with an initial uppercase letter for each word (e.g. "*North American Space Association*").	*he told John to go home* *He told* ***j****ohn to go home*	*hy het vir Johan gesê om huis toe te gaan* *hy het vir* ***j****ohan gesê om huis toe te gaan*
7. Punctuation	Punctuation marks were to be placed according to the rules and uses of the written language.	*he is a strong, healthy man* *he is a strong healthy man*	*hy is 'n sterk, gesonde man* *hy is 'n sterk gesonde man*
8. Hyphen	Hyphens were to be placed according to the rules and uses of the written language.	*he is the co-owner* *he is the* ***coowner***	*hy is die mede-eienaar* *hy is die* ***medeienaar***

9. Compound	Both invalid compound compositions as well as invalid compound decomposition were taken into account.	*there's nowhere else for it to go* *there's* ***no where*** *else for it to go*	*daar's nêrens anders waar dit kan heen nie* *daar's* ***nie êrens*** *anders waar dit kan heen nie*
10. Acceptable	If a respondent provided a valid spelling variant of a specific word the respondent was not penalised for the deviation from the gold standard, but the occurrence was categorised as an acceptable language error.	*he drank a lot of whisky* *he drank a lot of whiskey*	*hy het baie whisky gedrink* *hy het baie whiskey gedrink*
Protocol errors			
11. Capitalisation	Contrary to spelling conventions, words at the beginning of a sentence had to be written in lower case, except if the first word of a sentence was a name.	*the details of doing this properly are complex* ***T***_he details of doing this properly are complex_	*die detail betrokke om dit behoorlik te doen is ingewikkeld* ***D***_ie detail betrokke om dit behoorlik te doen is ingewikkeld_
12. Number	Respondents were instructed to write out ordinal numbers and numbers that make out part of a word instead of using digits (i.e. "*twelve*" and not "*12*").	*he is turning eighteen* *he is turning* ***18***	*hy word agtien* *hy word* ***18***
13. Abbreviation	If an abbreviations was heard in the audio recording, the letters of the abbreviation were to be written next to one another, separated with spaces, and in uppercase letters, even though this deviated from the conventional spelling (e.g. "*A T V*" (for all-terrain vehicle) instead of "*ATV*"). Respondents were instructed not to make use of abbreviations if it was not heard in the audio recording, i.e. if the word "*etcetera*" was heard, they were not permitted to write "*etc.*".	*an F B I case* *an* ***FBI*** *case*	*'n F B I-saak* *'n* ***FBI***_-saak_

14. Acronym	Acronyms were to be written entirely in uppercase letters, but not separated with spaces (e.g. “*NASA*”, “*UNISA*”).	*He is employed at NECCO* *He is employed at* ***Necco***	*hy werk by NECCO* *hy werk by* ***Necco***
15. Punctuation	Respondents were only allowed to use the following five punctuation marks: full stop, ellipsis, question mark, hyphen and comma.	*He said I will not be attending* *He said****:*** ***“****I will not be attending****”***	hy het gesê ek gaan nie bywoon nie hy het gesê**:** **“**Ek gaan nie bywoon nie**”**
16. White space	Respondents were only allowed to use single white spaces between words. They were penalised for multiple and excessive white spaces.	*in many cases, such as these* *in many cases , such as these*	*in baie gevalle, soos hierdie* *in baie gevalle , soos hierdie*
17. Terminator	Each utterance should end with one of the following three punctuation marks: full stop, ellipsis or question mark.	*sentences should end with a full stop.* *sentences should end with a full stop*	*sinne moet eindig met ’n punt.* *sinne moet eindig met ’n punt*
18. Acceptable	If a respondent was uncertain about the spelling of a name, he/she could add a question mark in brackets, *(?),* directly after the name followed by a space (e.g. “*Gadhafi(?) said that...”*). If this stipulation was followed correctly, the respondent was not penalised for the spelling error, but the occurrence was categorised as an acceptable protocol error.	*Jakarta is a city in Malaysia* ***Jacarta(?)*** *is a city in Malaysia*	*Jakarta is ’n stad in Maleisië* ***Jacarta(?)*** *is ’n stad in Maleisië*

Table 1: Categories of errors

For the step of manual classification of errors, all transcriptions were compared with the gold standard. The classification step is only performed if the comparison showed a difference.

2.4.6 Hypotheses

The hypotheses that were tested were as follows:

Task A: Lemmatisation

Time:

- The null hypothesis was that all groups are equal.
- The alternative hypothesis was that all groups are not equal.

For the variables accuracy, precision, recall, *f*-score, capitalisation errors, spelling errors, empty responses and total errors:

- The null hypothesis was that all groups are equal.
- The alternative hypothesis was that experts outperform novices and laymen (one-sided), with laymen and novices not equal.

For the variables of accuracy, precision, recall and *f*-score, higher scores imply better performance; for capitalisation errors, spelling errors, empty responses and total errors, lower scores imply better performance.

Task B: Orthographic transcription

Time:

- The null hypothesis was that all groups are equal.
- The alternative hypothesis was that all groups are not equal.

For the annotated errors:

- The null hypothesis was that all groups are equal.
- The alternative hypothesis was that experts outperform novices and laymen (one-sided), with laymen and novices not equal.

For all of these variables, lower scores imply better performance.

2.5 Results, analysis and interpretation

This section provides results as well as an analysis and interpretation of the results. The mean values and standard deviation of the variables in Task A and Task B are provided in Annexure D.

Analysis of the results was performed using four statistical models:

1. **One sample t-tests** were used to test if the mean score for the novice respondents was equal to the expert's score with a one-sided alternative (expert > mean novice). The same test was used to test if the mean score for laymen respondents was equal to the expert's score with a one-sided alternative (expert > mean laymen).
2. An **independent sample t-test** was performed as a parametric test to compare novices with laymen.
3. The **Wilcoxon rank sign test** was used as a non-parametric test to compare the expert to the novices as well as comparing the expert to the laymen.
4. The **Mann-Whitney test** was used as a non-parametric test to compare novices with laymen.

Bonferroni corrections were done on all *p*-values to compensate for multiple comparisons. For these four tests, a *p*-value smaller than 0.05 is considered as sufficient evidence that the result is **statistically significant**. **Box-plots** were used to determine outlying and extreme values. Should a respondent seem to be potentially problematic, the results of the respondent were removed before the other tests were performed.

In addition to reporting descriptive statistics, effect sizes are also determined. The effect size is independent of sample size and is a measure of practical significance. It can be understood as a large enough difference to have an effect in practice (Ellis & Steyn, 2003). Practical significance is reported as an additional measure. Whenever a result is not statistically significant, practical significance is not considered relevant and is not discussed. For the interpretation of the effect size of the t-tests, Cohen's *d*-value (Cohen, 1988) is used as a measure of practical significance. $d = \pm 0.2$ is considered a small effect (no practically significant difference), $d = \pm 0.5$ is considered a medium effect (practically visible difference) and $d = \pm 0.8$ is considered a large effect (practically significant difference). For the interpretation of the effect size of the Wilcoxon rank sign test and the Mann-Whitney test, an effect size correlation of $r = \pm\ 0.1$ is considered a small effect (no practically significant difference), $r = \pm\ 0.3$ is considered a medium effect (practically visible difference) and $r = \pm\ 0.5$ is considered a large effect (practically significant difference).

2.5.1 Task A: Lemmatisation of Afrikaans text data

2.5.1.1 Results

When comparing the time taken to complete the task (see Figure 1), the expert took 3,874 seconds (64.57 minutes), the novices an average of 3,863 seconds (64.36 minutes) and the laymen an average of 3,779 seconds (62.99 minutes). The differences between the groups were less than two minutes, indicating that neither group had a distinct advantage over the other in term of annotation time.

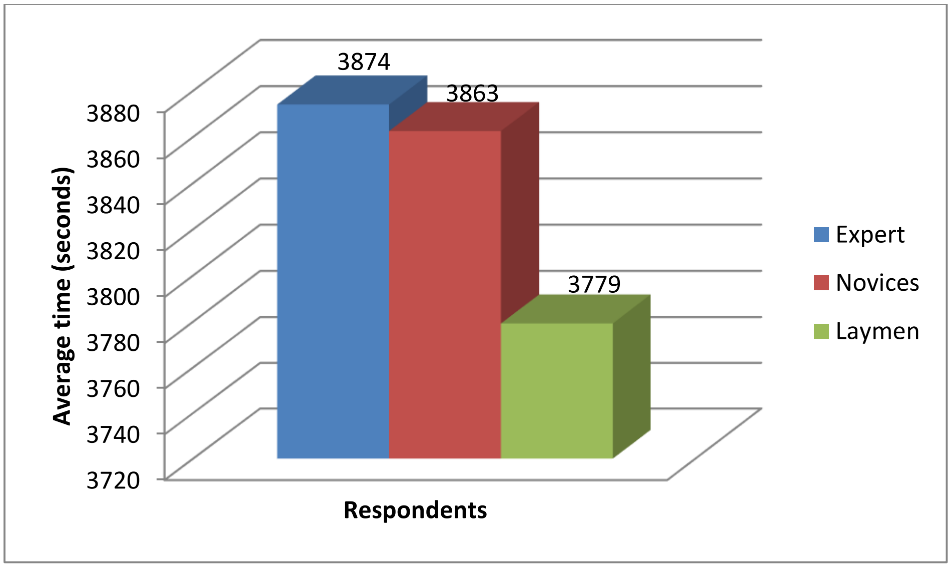


Figure 1: Average annotation time of lemmatisation

Table 2 shows accuracy, precision, recall and *f*-score of the different groups of respondents[21]. The expert achieved an accuracy 21.46% higher than the novices and 19.02% higher than the laymen. The *f*-score of the expert was 49.51% and 45.97% better than the novices and laymen respectively.

	Accuracy	Precision	Recall	*F*-score
Expert	97.10	94.78	80.74	87.20
Novice	75.64	30.36	50.59	37.69
Laymen	78.08	36.21	50.74	41.23

Table 2: Accuracy, precision, recall and *f*-score of the different groups of respondents

The low precision, recall and *f*-scores can mostly be attributed to (1) respondents not attempting to lemmatise words that needed to be lemmatised, for example "*opgevolg*" was left as is and not

[21] The mean values and standard deviation of the variables in Task A and Task B are provided in Annexure D.

lemmatised as "*opvolg*"; and (2) to derivations being lemmatised, for example "*benadering*" was lemmatised as "*benader*" instead of leaving the word as it originally appeared.

Analysing the errors made by the novice and laymen respondents showed no major difference between the numbers of errors made on the words to be lemmatised (591 vs. 548). On the words to be left unchanged, there was however a more pronounced difference. The novices erroneously tried to lemmatise 1,309 words, while the laymen attempted to lemmatise 952 words[22]. The protocol specifically specified that verbs with the prefixes *ge*-, *be*-, *her*-, *er*-, *ont*- and *ver*- were excluded in this task, and that words in these categories should not be lemmatised. The novices did, however, tend to remove these prefixes, and also incorrectly lemmatised derivations (e.g. "*betaling*" -> "*betaal*"), compounds (e.g. "*aansoek*" -> "*soek*"), and pseudo forms (i.e. words that are orthographically similar so inflectional forms; e.g. "*anders*" -> "*ander*") more frequently than the laymen. One reason for this might be that the laymen followed the protocols stricter than the novices as they are not routinely exposed to linguistics. The novices might have been under the impression that they possess the knowledge to perform the task without studying the protocol and consequently made these types of errors. Apart from the incorrect lemmatisation of words, the difference in results achieved by the three groups of respondents can also be attributed to the number of capitalisation errors, spelling errors and to a lesser extent the instances where the respondent failed to provide an answer. Figure 2 shows the average capitalisation errors, spelling errors and empty responses made by the different groups of respondents.

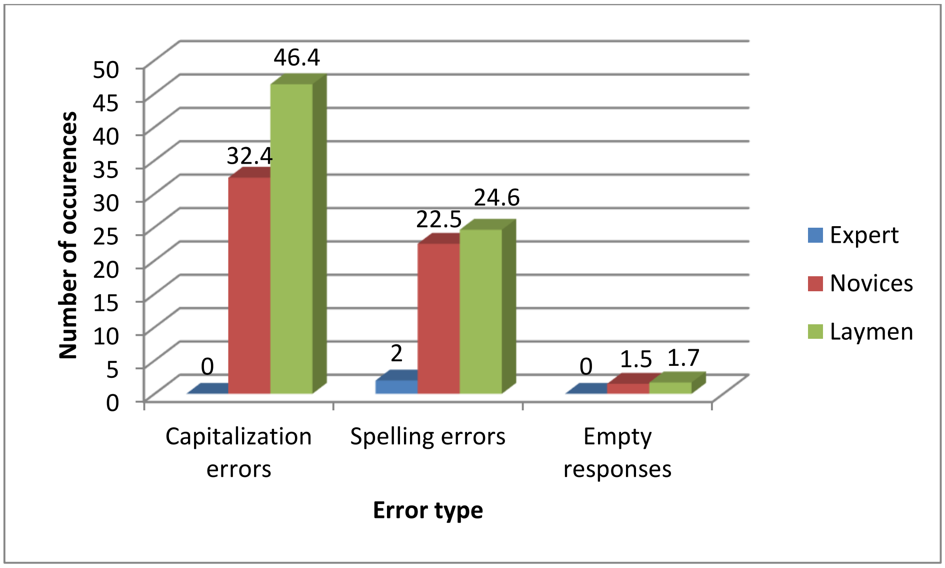


Figure 2: Capitalisation errors, spelling errors and empty responses in Task A (lemmatisation)

[22] Capitalisation and spelling errors were excluded in these comparisons.

The expert respondent made only two spelling errors and no capitalisation errors or empty responses. The novices and laymen made a total average of 56.4 and 72.7 errors respectively. It is apparent from these results that the expert outperforms both the novices and laymen in terms of the quality of the annotations.

2.5.1.2 Statistical analysis and interpretation

Although the results in the previous section show that the expert outperformed both the novices and laymen, further investigations were conducted to determine the significance of these differences. First, the data from the expert was compared with data from novices and laymen in order to establish if all groups were equal in terms of time and/or overall performance. Thereafter, novices and laymen were compared to determine if there was a difference between the two groups in terms of time and/or overall performance.

In Table 3 the *p*-values and *d*-values from the one sample t-test, as well as the *p*-values and *r*-values from the Wilcoxon rank sign test, are reported for the comparison between the expert and novices as well as between the expert and laymen. Bonferroni corrections were performed on all *p*-values to compensate for multiple comparisons.

The one sample t-test showed that no statistical significant differences were present when comparing the time needed to complete the task between the expert and novices or expert and laymen. The accuracy, precision, recall and *f*-score achieved by the expert were statistically significantly better when compared to the scores of novices and laymen, with all $p < 0.001$. The *d*-values and *r*-values also illustrated that there is a practically significant difference in the scores of the expert compared to scores of the novices and laymen.

The data of the expert also contained statistically significantly fewer capitalisation errors (expert vs. novices, $p = 0.005$; expert vs. laymen, $p < 0.001$) and spelling errors (expert vs. novices, $p < 0.001$; expert vs. laymen, $p = 0.004$). The *d*-values and *r*-values also showed a practically significant difference. The empty responses showed no statistically significant difference for expert vs. novices ($p = 0.107$) and expert vs. laymen ($p = 0.095$).

	Expert vs. Novices				**Expert vs. Laymen**			
	One sample t-test		**Wilcoxon rank sign test**		**One sample t-test**		**Wilcoxon rank sign test**	
	Bonferroni corrected *p*	**Effect size (*d*)**	**Bonferroni corrected *p***	**Effect size (*r*)**	**Bonferroni corrected *p***	**Effect size (*d*)**	**Bonferroni corrected *p***	**Effect size (*r*)**
Time (s)	1.000	-0.033	1.000	0.048	0.965	-0.152	1.000	0.048
Accuracy	< 0.001	-5.031	0.008	0.886	< 0.001	-4.613	0.008	0.886
Precision	< 0.001	-5.860	0.008	0.886	< 0.001	-7.923	0.008	0.886
Recall	< 0.001	-2.418	0.008	0.886	< 0.001	-2.445	0.008	0.886
F-Score	< 0.001	-4.134	0.008	0.886	< 0.001	-5.648	0.008	0.886
Capitalisation errors	0.005	-1.782	0.018	0.666	< 0.001	-2.656	0.012	-0.800
Spelling errors	< 0.001	-2.824	0.008	0.816	0.004	1.249	0.008	0.886
Empty responses	0.107	-0.914	0.065	0.416	0.095	-0.946	0.042	-0.428

Table 3: Analysis of expert vs. novices and expert vs. laymen

The Wilcoxon rank sign test showed statistically and practically significant differences on all variables except time and empty responses between the data annotated by the expert and novices. The comparison of the expert and laymen data showed statistically and practically significant differences on all variables except time.

When comparing novices with laymen, the independent sample t-test showed no statistically significant differences (see Table 4). The Mann-Whitney test showed no statistically or practically significant differences in the comparison of the data annotated by the novices and laymen.

	Novices vs. Laymen			
	Independent sample t-test		**Mann-Whitney**	
	Bonferroni corrected *p*	**Effect size (*d*)**	**Bonferroni corrected *p***	**Effect size (*r*)**
Time (s)	1.000	0.133	1.000	0.161
Accuracy	0.639	-0.689	0.922	-0.296
Precision	0.556	-0.529	0.819	-0.245
Recall	1.000	-0.012	1.000	-0.034
F-Score	1.000	-0.296	1.000	-0.127
Capitalisation errors	0.691	-0.555	0.391	-0.267
Spelling errors	1.000	0.162	1.000	0.008
Empty responses	1.000	-0.082	1.000	-0.041

Table 4: Analysis of novices vs. laymen

In order to determine if a combination of the non-expert data (i.e. data from novices and laymen) can result in data of equal quality to that of the expert, we combined all the data from the non-experts. The data was combined using simple majority voting. Twenty new datasets were compiled starting with one respondent and randomly adding the data of another respondent for each subsequent dataset. The combined data was evaluated on accuracy.

Figure 3 represents the accuracy of the combined datasets of non-experts in relation to the expert. The individual scores of the non-experts are also plotted. Although the combined data is more accurate than the individual data, the combined accuracy never reached the accuracy achieved by the expert. The closest score was reached after combining the data of seven respondents, but the combined score was still 8% lower than that of the expert.

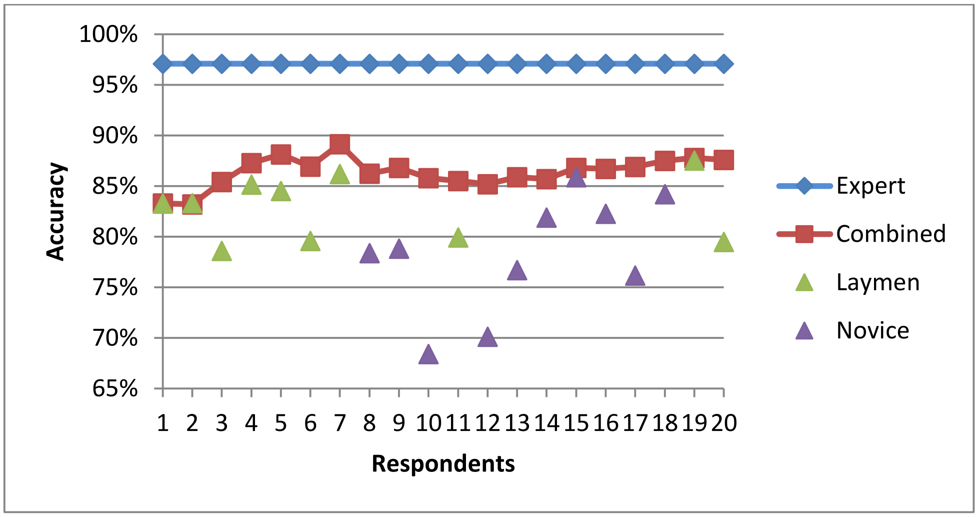


Figure 3: Combined accuracy of datasets from non-experts in relation to accuracy of expert

Next, the datasets from the ten non-expert respondents achieving the highest accuracy were combined. Figure 4 shows the accuracy of the combined datasets of the ten best respondents, with individual scores plotted. As with the first combined datasets using all twenty respondents, the combined accuracy is better than the individual accuracy, but still does not reach the accuracy of the expert. The closest score was reached after combining the data of nine respondents, but the combined score was still about 6% lower than that of the expert. It seems that for lemmatisation, untrained non-experts are not able to annotate data with similar accuracy to experts.

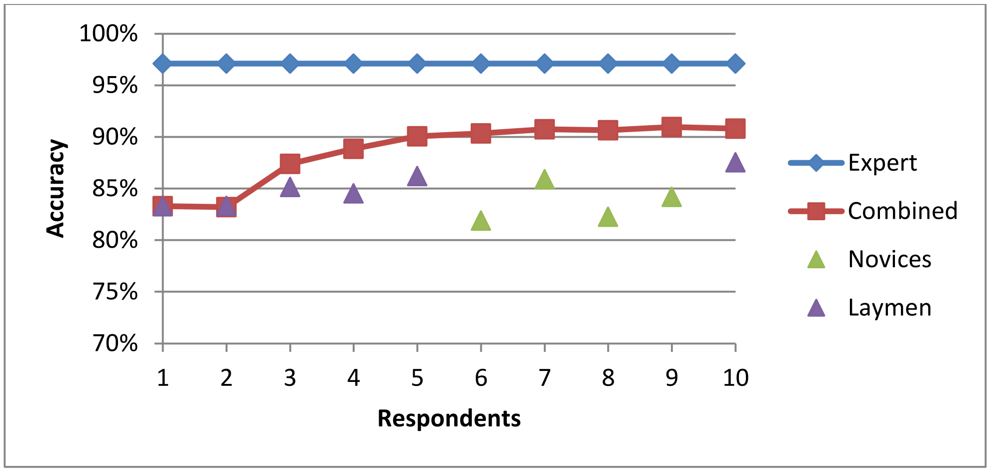


Figure 4: Combined accuracy of ten best datasets from non-experts in relation to accuracy of expert

2.5.2 Task B: Orthographic transcription of Afrikaans audio data

2.5.2.1 Results

When comparing the time taken to complete the task (see Figure 5), the expert took 4,058 seconds (67.63 minutes), the novices an average of 3,939 seconds (65.64 minutes) and the laymen an average of 4,689 seconds (78.14 minutes). The differences between the expert and novices were less than two minutes, indicating that neither group has a distinct advantage over the other in terms of annotation time. On average, the laymen took 10.5 minutes longer than the expert to complete the task.

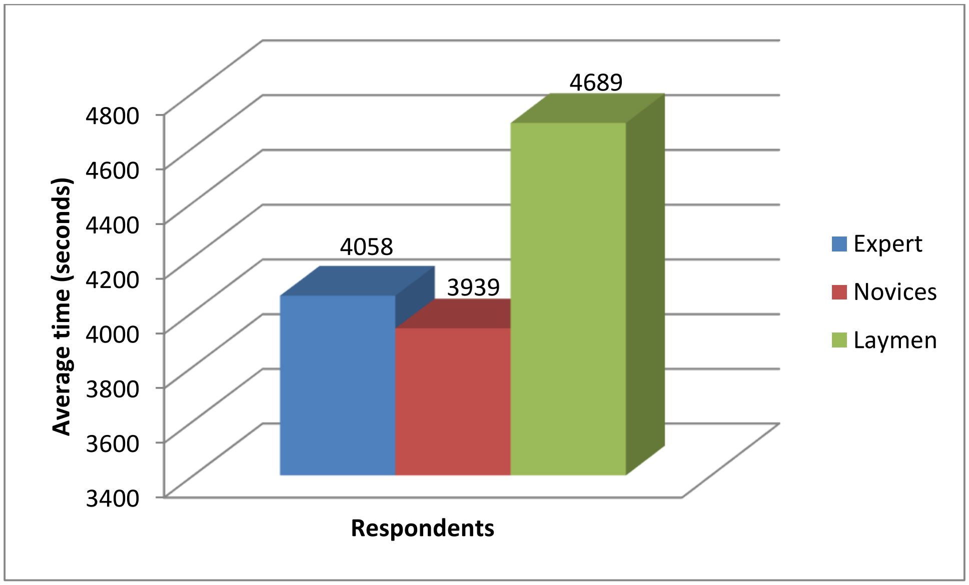


Figure 5: Average time of orthographic transcriptions

The errors found in orthographic transcriptions are classified in one of three categories: transcription errors, language errors and protocol errors (see 2.4.5). Figure 6 illustrates the average errors in these categories and the total average errors made by the three types of respondents.

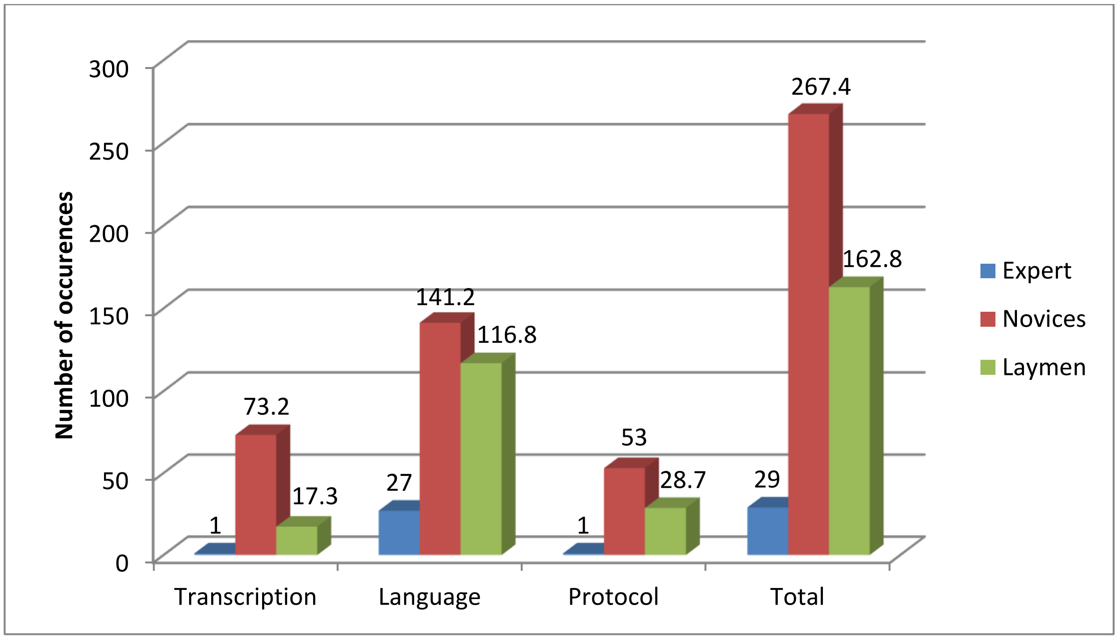


Figure 6: Total annotated errors made by the different groups of respondents in Task B (orthographic transcription)

Table 5, Table 6 and Table 7 provide a breakdown of the average transcription, language and protocol errors. From the breakdown of transcription errors, it is evident that the expert data contained fewer transcription errors than the data from the novices or laymen. The expert made only one substitution error. It is interesting to note that the laymen made fewer transcription errors than the novices.

	Insertions	**Deletions**	**Substitutions**	**Transpositions**
Expert	0.00	0.00	1.00	0.00
Novice	12.00	41.70	19.30	0.20
Laymen	3.20	6.30	7.80	0.00

Table 5: Breakdown of transcription errors

The breakdown of language errors again show that the expert made fewer language errors than the novices and laymen. The differences in language errors were even more prominent than the differences in the transcription errors, and the language errors contributed to the majority of all annotated errors in the data. The expert only misspelled six words while novice respondents made an average of 69.5 spelling errors and laymen made an average of 46.9 spelling errors. Once again, the laymen made fewer language errors than the novices, even though one would expect the novices to be better suited to the task given their linguistic background.

	Spelling error	Capitalisation	Punctuation	Hyphen	Compound
Expert	6.00	3.00	14.00	1.00	3.00
Novices	69.50	21.00	23.90	8.00	18.80
Laymen	46.90	18.70	22.00	10.70	18.50

Table 6: Breakdown of language errors

The breakdown of protocol errors showed a similar trend to the breakdown of the transcription and language errors. Yet again the expert outperformed both the novices and laymen, and the laymen outperformed the novices.

	Capitalisation	Number	Abbreviation	Acronym	Punctuation	White Space	Terminator
Expert	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Novice	11.70	4.30	4.70	1.20	7.40	13.40	10.30
Laymen	6.20	4.40	3.50	1.60	6.20	4.40	2.40

Table 7: Breakdown of protocol errors

The differences between the novices and laymen in the category protocol errors might be attributed to the fact that the protocol included stipulations that deviated from the standard written orthography of the language, and from the results it seems as if the novices did not follow these stipulations stringently, but rather conformed to the language rules. Besides the difference in protocol errors, the most prominent difference between the novice and laymen data was in the category of transcription errors, with an average of 73.2 errors for novices, and 17.3 errors for the laymen. The box-plots used in the statistical analysis in the following section, show that the novice data contained outlying and extreme values. When these values were removed from the transcription errors, the difference between the two groups was greatly reduced, and showed an average of 34.08 errors for novices and 17.3 errors for laymen[23]. One final factor that might have influenced the performance is the time taken to complete the task by the different groups. The novices completed the task on average 12.5 minutes faster than the laymen, which might indicate that they rushed to complete the task, and as a result made extra errors.

[23] As stated in 2.5, outlying and extreme values were removed before performing the statistical analysis in the following section, where the significance of the difference between the novices and laymen will be determined.

2.5.2.2 *Statistical analysis and interpretation*

As with the results of Task A, further investigations were conducted to determine the significance of the differences. In Table 8 below, the *p*-values and *d*-values from the one sample t-test as well as the *p*-values and *r*-values from the Wilcoxon rank sign test were reported for the comparison between the expert and novices as well as between the expert and laymen. Once again, Bonferroni corrections were performed on all *p*-values to compensate for multiple comparisons.

	Expert vs. Novices				**Expert vs. Laymen**			
	One sample t-test		**Wilcoxon rank sign test**		**One sample t-test**		**Wilcoxon rank sign test**	
	Bonferroni corrected *p*	**Effect size (*d*)**	**Bonferroni corrected *p***	**Effect size (*r*)**	**Bonferroni corrected *p***	**Effect size (*d*)**	**Bonferroni corrected *p***	**Effect size (*r*)**
Time (s)	0.829	-0.200	1.000	0.177	0.148	0.580	0.418	0.467
Total Errors	< 0.001	1.817	0.008	0.886	0.001	1.634	0.008	0.886
Transcription errors								
Insertions	0.031	0.890	0.008	0.886	0.013	1.060	0.027	0.894
Deletions	0.099	0.660	0.008	0.886	< 0.001	2.010	0.008	0.886
Substitutions	0.020	0.970	0.008	0.886	< 0.001	2.020	0.008	0.886
Transpositions	0.252	0.470	0.270	0.949	n/a	n/a	n/a	n/a
Language errors								
Spelling error	0.001	1.750	0.008	0.886	0.030	0.890	0.008	0.886
Capitalisation	< 0.001	2.320	0.008	0.886	< 0.001	3.220	0.008	0.886
Punctuation	0.004	1.290	0.010	0.854	0.011	1.090	0.012	0.889
Hyphen	0.001	1.560	0.008	0.886	< 0.001	1.930	0.008	0.886
Compound	0.001	1.590	0.010	0.854	0.021	0.960	0.008	0.886
Protocol errors								
Capitalisation	0.056	0.770	0.012	0.889	0.096	0.670	0.008	0.886
Number	0.023	0.940	0.042	0.899	0.246	0.480	0.065	0.905
Abbreviation	< 0.001	1.820	0.018	0.891	0.002	1.480	0.018	0.891
Acronym	0.004	1.310	0.027	0.894	0.002	1.490	0.018	0.891
Punctuation	0.017	1.010	0.018	0.891	0.036	0.860	0.012	0.889
White space	0.006	1.230	0.008	0.886	0.061	0.760	0.027	0.790
Terminator	0.075	0.710	0.012	0.889	0.116	0.630	0.027	0.894

Table 8: Analysis of expert vs. novices and expert vs. laymen

The one sample t-test as well as the Wilcoxon rank sign test of expert vs. novices and expert vs. laymen illustrated that there were no statistically significant differences present in the time needed to complete the task. The one sample t-test showed a statistical significant difference on total annotated errors (expert vs. novices, $p < 0.001$; expert vs. laymen, $p = 0.001$) was present. The expert vs. novices showed statistically significant differences on the category transcription errors on insertions (*p* = 0.031) and substitutions (*p* = 0.020). The expert vs. laymen showed statistically significant differences on all sub-categories of transcription errors, except on transpositions as the standard deviation was zero and thus not applicable. All sub-categories that showed statistical significance also showed practical significance on *d*-values and *r*-values.

In the category language errors, the one sample t-test of expert vs. novices and expert vs. laymen show a significant difference on spelling errors (expert vs. novices, *p* = 0.001; expert vs. laymen, *p* = 0.030), capitalisation (expert vs. novices, *p* < 0.001; expert vs. laymen, *p* < 0.001), punctuation (expert vs. novices, *p* = 0.004; expert vs. laymen, *p* = 0.011), hyphens (expert vs. novices, *p* = 0.001; expert vs. laymen, *p* < 0.001) and compounds (expert vs. novices, *p* = 0.001; expert vs. laymen, *p* = 0.021). The *d*-value and *r*-value show a practically significant difference on all these sub-categories as well.

In the protocol category, the quality of transcription performed by the expert was almost perfect with only one erroneous white space present in the data. The novices made an average of 53 protocol errors and the laymen made an average of 28.7 protocol errors. The breakdown of errors between expert and novices shows statistically significant differences on number, abbreviation, acronym, punctuation, and white space, and shows statistically significant differences between expert and laymen on abbreviation, acronym and punctuation. The *d*-values and *r*-values show practically significant differences on these results as well.

When comparing novices with laymen, an analysis of analyses shows that no statistically significant differences were present on time taken for transcription or total annotated errors (see Table 9). Although the results in 2.5.2.1 show that the laymen made fewer transcription, language and protocol errors than the novices, the analysis of the differences shows that these differences are not statistically or practically significant. We can therefore conclude that, for our respondents, it is not beneficial to use novices instead of laymen for the task of orthographic transcription of audio data for Afrikaans.

	Novices vs. Laymen			
	Independent sample t-test		Mann-Whitney test	
	Bonferroni corrected *p*	**Effect size (*d*)**	**Bonferroni corrected *p***	**Effect size (*r*)**
Time (s)	0.218	-0.692	0.724	-0.262
Total Errors	0.139	0.797	0.246	0.389
Transcription errors				
Insertions	0.180	0.650	0.070	0.507
Deletions	0.278	0.562	0.012	0.642
Substitutions	0.223	0.608	0.596	0.287
Transpositions	0.453	0.474	1.000	0.161
Language errors				
Spelling error	0.710	0.494	0.209	0.406
Capitalisation	1.000	0.296	1.000	-0.008
Punctuation	1.000	0.248	1.000	0.177
Hyphen	0.666	-0.536	0.337	-0.355
Compound	1.000	0.019	1.000	0.127
Protocol errors				
Capitalisation	1.000	0.363	1.000	0.000
Number	1.000	-0.011	1.000	0.152
Abbreviation	0.880	0.464	0.558	0.296
Acronym	1.000	-0.372	1.000	-0.177
Punctuation	1.000	0.163	1.000	0.042
White space	0.057	0.893	0.038	0.558
Terminator	0.334	0.548	0.094	0.482

Table 9: Analysis of novices vs. laymen

As with Task A, we combined all the data from the non-experts to determine if a combination of the non-expert data can result in data of equal quality to that of the expert. The data was combined in two ways: a case sensitive comparison, and by converting the data to lower case before combining the data; this was done in order to improve the likelihood of finding matching transcriptions. In both methods, we used simple majority voting to decide what transcription should be included in the new dataset. The combined data was evaluated on total annotated errors present in the data.

Figure 7 represents the total annotated errors of the combined datasets of non-experts in relation to the expert, as well as the individual scores of the non-experts. Similar to Task A, the combined data is more accurate than the individual data, but the combined accuracy never reaches the accuracy achieved by the expert. The closest score was reached after combining the lower case data of nineteen respondents, but the combined score still contained 99 more errors than the data of the expert.

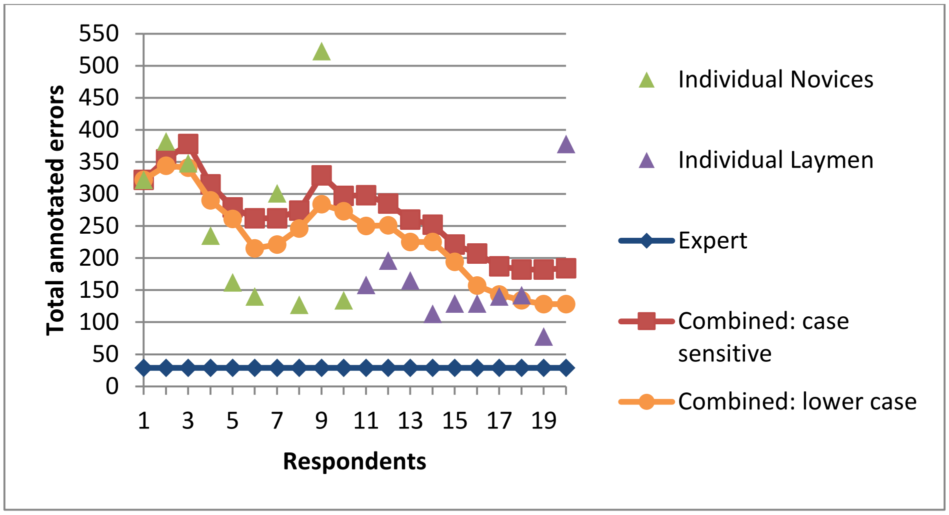


Figure 7: Total annotated errors in the combined datasets from non-experts in relation to expert

Next, the datasets from the ten best non-expert respondents were combined. Figure 8 shows the total annotated errors present in the combined datasets, with individual scores plotted. As with the first combined datasets using all twenty respondents, the combined data contained less errors than the individual data, but still does not reach the quality of the expert.

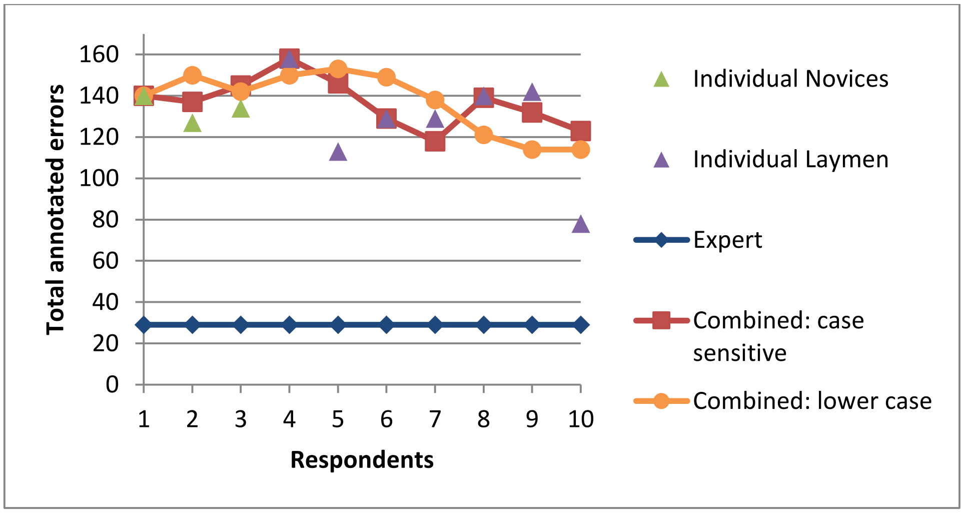


Figure 8: Total annotated errors of ten best datasets from non-experts in relation to expert

The closest quality was reached after combining the lower case data of nine respondents, but the combined data still contained 85 more errors than the data of the expert. Based on these results and analysis of the comparison of the expert and novices as well as between the expert and laymen, it seems that, for the task of orthographic transcription of Afrikaans audio data, results obtained using untrained non-experts are not comparable to results of experts.

2.6 Conclusion

The aim of this chapter was to establish if non-experts are suitable for the task of annotating linguistic data, in order to address our research questions:

1. Can similar results be obtained using experts and non-experts for annotation of resource-scarce languages?
2. If comparable results can be obtained using non-experts, is it beneficial to use novice annotators instead of laymen?

We provided an overview of previous projects that used non-experts to annotate data, and from these studies it was apparent that non-experts are indeed mostly suitable. The shortcomings of these studies were however that they were mainly focused on mainstream languages and basic linguistic tasks. Because it was not clear if the same approach could be used for resource-scarce languages and more complex tasks, we conducted an experiment using a crowdsourcing approach. We determined that crowdsourcing might not be a viable approach for annotation of linguistic data for Afrikaans, and suggested that the same might be true for other resource-scarce languages as only a few of these languages form part of the crowdsourcing community. No suitable workforce was available, mainly due to the demographics of users of crowdsourcing software, limited internet access and payment structure.

We nonetheless decided to test the suitability of non-experts practically by sourcing respondents to complete the tasks. We sourced two experts (one per task) and two groups of non-experts; twenty novices (undergraduate students studying for a bachelor's degree with the subject Afrikaans included in their curriculums) and twenty laymen (people who had never studied Afrikaans at tertiary level) to determine if these different skill levels might influence performance.

Next, we conducted systematic experiments where variables that could influence the results of the experiments were kept constant, ensuring that the only variable on the tasks was the skill level of the respondents. The respondents were tasked with providing lemmas for 1,000 words (Task A) and providing orthographic transcriptions of six minutes of audio data (Task B). No statistically significant differences were visible in the comparison of average time taken to complete Task A or Task B between any of the groups.

For the task of lemmatisation, the expert achieved an accuracy of 97%, while the novices and laymen achieved 75% and 78% respectively. The accuracies of both groups were statistically and practically significantly lower than the accuracy achieved by the expert. No statistically significant difference was present when comparing the data of the novices and laymen. We also combined the data of the novices and laymen by means of voting, to see if we could reach accuracy comparable to that of the expert, but the best combined accuracy was still 6% lower than that of the expert.

For the task of orthographic transcription of audio data, the expert made a total of 29 errors, while the novices made an average of 267.4 errors and the laymen 162.8 errors on average. A breakdown of the error types into transcription, language and protocol errors showed statistically significant differences between the expert and novices, and the expert and laymen. No statistically significant difference between the novices and laymen was evident.

From the results it was evident that the experts outperformed the non-experts on both tasks, and that the differences in performance were significant. We concluded that results obtained using untrained non-experts are not comparable to results of experts. Analysis of the comparison between novices and laymen for both tasks showed no statistically significant differences, and we can therefore conclude that it is not beneficial to use novices instead of laymen for the task of lemmatisation or for the task of orthographic transcription of audio data for Afrikaans. We performed error analysis on the novice and laymen data in order to determine why the laymen outperformed the novices, and speculated that one reason might be that the laymen followed the protocols stricter than the novices as they are not routinely exposed to linguistics. A second reason can be attributed to the outlying and extreme values present in the transcription data of the novices.

It is important to note that none of the respondents received any training, and this could be a major influence on the accuracy of the annotations. In the following chapter, we explore the effect of annotation environments on data annotation by using trained novices to perform the same tasks in different software environments. The group that used the same software as in this chapter (i.e. *CrowdFlower*) achieved an accuracy of 90% for lemmatisation and, on average, only made 91.8 errors in the orthographic transcriptions. When taking this into account, it seems as if non-experts might be suitable for annotation tasks, but they need a sufficient level of training. This will be discussed in more detail in Chapter 5.

3 Chapter 3: The effect of software environments on data annotation

3.1 Introduction

The focus of this chapter is on the user interface that the annotator uses, and the influence it has on the annotator's performance. As the need for annotated data grows, the need for suitable annotation environments also increases. The main motivations for the development of tailor-made annotation environments are (1) to fill a need that existing software packages cannot, (i.e. the available software lacks some functionality, such as a specific annotation framework, support for custom tagsets, or incompatibility with other software); and (2) it is assumed that annotation can be performed faster and/or more accurate in tailor-made software.

The following section provides an overview of general-purpose software, domain-specific software and tailor-made software as well as a brief description of existing software developed for use in linguistic annotation of data. In section 3.4, experiments in seven software environments are described, and thereafter the experiments are evaluated in terms of time needed to perform the task and the quality of the data. The time of developing tailor-made software will also be weighed against any savings in annotation time in order to determine whether the additional time can be justified (see 3.6). The findings are not relevant only to resource-scarce languages, as the functionalities incorporated in tailor-made software can also contribute directly to the success of annotation projects for mainstream languages.

3.2 Literature survey

In any HLT-related annotation project, one of the crucial choices a project faces is what software to use for the purpose of data annotation. Data annotation software can be summarised in three broad categories:

1. **General-purpose software**: These software environments are developed to assist users in accomplishing simple computer-related tasks, for example word processing. These software environments are generally well-supported and implemented by established software vendors, and tend to have mature feature sets. The features included in such software are aimed at a general client base and developers aim to satisfy the needs of most users. Although general-purpose software usually lacks some features that could be beneficial in annotation projects, many annotation projects use these software environments for annotation, especially

spreadsheets such as *Microsoft Excel*, *Gnumeric*[24], *Quantrix*[25], *KCells*[26], etc. Other examples of these software packages include word processors, databases, and multimedia software. Crowdsourcing software such as *Mechanical Turk*[27], *CrowdFlower*[28], *BizReef*[29], *Elance*[30], *Freelancer*[31] and *SamaSource*[32]is also seen as general-purpose software as it only provides an interface to facilitate response from users, i.e. it makes provision for someone to post a task (e.g. "*Is there a person in this picture?*", "*What is the lemma of the provided word?*", "*Provide the transcription of the audio file.*", etc.) and receive a response from a respondent.

2. **Domain-specific software**: These software environments are developed for data annotation in a broad domain – for example audio transcription in general – and might include features relevant to a specific task, but are not specifically developed according to the requirements of a project. It is often the case that domain-specific software is initially developed as tailor-made software for a project, but at some stage the software is made available to outside users or other projects. In order to meet the needs of the increased user base, additional features are included, but these features tend to be more general-purpose in nature. Examples include *Praat* (Boersma, 2002), *MMAX* (Müller & Strube, 2001) and *Callisto* (Day *et al.*, 2004).

3. **Tailor-made software**: These software environments (also known as bespoke or custom software) are developed to solve specific needs of a specific client. These solutions often offer the greatest flexibility and are developed according to the requirements of a specific project. The features included in such software are aimed at improving or simplifying the annotation process and are relevant and applicable to the specific task. Examples include the *ITUtreebank annotation tool* (Eryigit, 2007), *Quick Annotator* (Strassel *et al.*, 2005) and *PALinkA* (Orasan, 2003).

[24] http://projects.gnome.org/gnumeric

[25] www.quantrix.com

[26] www.koffice.org

[27] www.mturk.com

[28] www.crowdflower.com

[29] www.bizreef.com

[30] www.elance.com

[31] www.freelancer.com

[32] www.samasource.org

The following discussion concerns the main advantages and disadvantages of these software categories relevant to the task of annotation. Other dimensions of software, such as design and layout, server-based versus client-based, and operating system compatibility were not taken into consideration in this discussion. For standard and in-depth discussions, comparisons of different software categories or these and other omitted dimensions, see Bergquist *et al.* (2011), Berntsson-Svensson and Aurum (2006), Fuggetta (2003), Green (2011), Li *et al.* (2009), McKinney (2001), Stamelos *et al.* (2003), Vigder *et al.* (2010) and Voas (1998) among others.

When considering the best choice of software for a particular task or project, time and budget constraints are often some of the deciding factors. In the case of **general-purpose software**, annotators usually have the software installed already – for example *Microsoft Office* or *Open Office* – or, as with crowdsourcing software, have access to the software without the need to install it. Using software packages already available to annotators has the benefit that there is no additional need for development time or cost. If the software needs to be purchased, it is typically reasonably priced as the cost is distributed between large numbers of buyers or free if Open Source. Annotators might also be familiar with the software, although they might only have used it for purposes other than annotation. This reduces the need for training of the annotators, as they need only be trained in the annotation task to be performed. One detrimental effect of using general-purpose software is that data will have to be adapted per project, according to the capabilities of the software. For example, if a project entails lemmatisation of a list of words, the choice of software might be an available spreadsheet package. The data could be represented in one column and the protocol will stipulate that the annotator is to provide the lemma in the adjacent column. Thus, the data will need to be converted to a suitable file format, such as comma-separated values (*.csv*), and will need to be reconverted to the original file format after the task has been completed.

If a project determines that general-purpose software lacks some important feature that might be beneficial to the project, it might be decided to use **domain-specific software**. Although there is no need to provide for development time or cost as the software is readily available, one delay might be finding suitable software for the project, as an abundance of software that could be used for the task might be available. Time might have to be spent on evaluating different software applications in order to determine which is most suitable for the task. Even if suitable software is found, the software will not necessarily include all relevant features that could benefit the task. Another major challenge in using domain-specific software is that insights about the structure of the annotations (i.e. stand-off annotation, inline annotation, embedded annotation, etc.) and data formats (i.e. plain text, markup languages (SGML/HTML/XML), etc.), are often buried in coding manuals. Although the software is usually free, as most are available Open Source, some packages are quite expensive since they include a wide range of task specific features.

If a project determines that the available software is not suited to its needs, it might be decided to develop **tailor-made software**. Since tailor-made software is developed for a specific task or project, it can accommodate particular preferences and expectations. Tailor-made software is developed to fill a need that existing software packages cannot, i.e. available software lacks some crucial features. Omitted features could include support for custom tagsets, incompatibility with a preferred annotation structure, incompatibility with a specific operating system, etc. Additional features that are relevant to the task and that could benefit the project can also be incorporated into tailor-made software. These additional features could include some form of automatic protocol checking, spelling checking, automated backup of data, etc. Although tailor-made software might prove invaluable for a task, the development can be time-consuming and expensive as software development usually includes system analysis, design, testing, operations and maintenance, and the cost is usually carried by a single client or project.

Despite the additional time and cost associated with the development of tailor-made software, the number of tailor-made annotation software products has increased over the past couple of years. This increase can be attributed to the growing need for annotated data. Müller and Strube (2001) support this notion and refer to the motivation for the development of *MMAX* as: “The growing need for the annotation of multi-modal corpora on the one hand and the lack of tools that are productively usable for this task on the other”. Bird and Harrington (2001) confirm this by stating that the growth in the use of corpora has not been matched by the development of a standard set of tools for creating, editing, annotating and querying corpora. As a result, many projects develop their own systems for corpus annotation and analysis, because existing tools are ill-equipped to cope with the increasing size and range of applications for which corpora are constructed.

The importance of developing suitable tailor-made annotation software is evident from regular conferences and workshops, for example the *Linguistic Annotation Workshop* (*LAW*)[33], *Language Resources and Evaluation* (*LREC*)[34], *International Conference on Computational Linguistics* (*COLING*)[35],

[33] www.ling.uni-potsdam.de/acl-lab/LAW-07.html

[34] www.lrec-conf.org

[35] www.nlp.shef.ac.uk/iccl

Interspeech[36], and *Text Encoding Initiative (TEI) Special Interest Group on Tools*[37], which dedicate special interest groups and sessions to software and the annotation of linguistic resources. The large quantity of software presented at these and other conferences and workshops further illustrates the importance of, and need for, suitable annotation software. Available software includes domain-specific as well as tailor-made software and might be aimed at only one specific task, or designed to be able to accommodate more than one task. Table 10 provides some examples of available annotation software and a brief description of the task(s) the software is intended for.

Product	**Task**
AnCoraPipe (Bertran *et al.*, 2008)	Semantic role labelling and annotation of named entities and co-reference.
Annotate (Plaehn & Brants, 2000)	Part-of-speech tagging and syntactic parsing.
ANVIL (Kipp, 2008)	Adding structured human annotations to digital video material.
Brandeis Annotation Tool (Verhagen, 2010)	Temporal relation annotation.
Callisto (Day *et al.*, 2004)	Various tasks, such as named entity tagging, event and temporal expression tagging, etc., by implementing custom Java modules. Also supports multilingual annotations.
EXMARaLDA (Schmidt & Wörner, 2008)	Transcription editor, corpus management tool and corpus query tool.
GATE (Cunningham *et al.*, 2002)	Framework and graphical development environment for creating and deploying language engineering components and resources.
ITUtreebank annotation tool (Eryigit, 2007)	Morphological analysis, part-of-speech tagging and syntactic parsing.
LabelMe (Russell *et al.*, 2008)	Annotation of images.
LinguaStream (Bilhaut & Widlöcher, 2006)	Part-of-speech, syntax, semantics, discourse and statistical tagging.
MAMI (Anguera & Oliver, 2008)	Annotation and searching for digital photos on a camera phone via speech input.
MMAX (Müller & Strube, 2001)	Annotation of multi-modal corpora.
NOOJ (Silberztein, 2005)	Morphological parsing, lexical parsing and local grammars.
PALinkA (Orasan, 2003)	Discourse annotation.

[36] www.isca-speech.org

[37] www.tei-c.org/Activities/SIG/Tools

Phrase Detectives (Chamberlain *et al.*, 2008)	Anaphoric annotation.
Praat (Boersma, 2002)	Analysis, synthesis, manipulation and transcription of audio data.
RSTTool (O'Donnell, 2000)	Markup of rhetorical structure of text.
TASX-annotator (Milde & Gut, 2002)	Annotation of empirical language data (video and audio material).
The MATE Workbench (McKelvie *et al.*, 2001)	Annotation of speech dialogues.
Transcriber (Barras *et al.*, 2001)	Manual segmentation and transcription of long duration broadcast news recordings, including annotation of speech turns, topics and acoustic conditions.
Wavesurfer (Sjölander & Beskow, 2000)	Viewing, editing, and labelling of audio data.
XDMLTool (Hardy *et al.*, 2003)	Annotating transcribed dialogues according to semantic, functional and stylistic characteristics.
XTrans (Maeda *et al.*, 2006)	Multilingual, multi-channel transcription tool.
Yawat and *Kwipc* (Germann, 2007)	Word and phrase alignment of parallel text and word pairs.
ZoneTag (Ahern *et al.*, 2006)	Media annotation of photographs via context-based tag suggestions.

Table 10: Annotation software and intended tasks

Software for annotation of data for specific languages is also developed. This is often due to an extended character set used by a specific language (for example diacritics in a language such as Tshivenda, e.g. ḽ, ṱ, ḓ, ṋ and ṅ), or non-Latin characters used by a language such as Mandarin Chinese, Levantine Arabic, or conversational Czech. Some structures that cannot be accommodated by existing software might also exist within the language. For example in Chinese, unlike English, semantic and syntactic boundaries often do not coincide (Strassel *et al.*, 2005). This is problematic for the annotation of discourse units with available software. Two examples of software for specific languages are *Quick Annotator* (Strassel *et al.*, 2005) which was developed for meta-data annotation of audio data for Czech, and the *ITUtreebank annotation tool* (Eryigit, 2007) which was designed to accommodate the particular morphological structure of Turkish.

In addition to the need for suitable annotation environments, time and money are also often invested in the development of tailor-made software as it is assumed that annotation can be performed faster and/or more accurately in tailor-made software than in domain-specific and general-purpose software. This assumption also implies that it simplifies workflow and further processing by implementing standards and protocols. Bertran *et al.* (2008) state that *AnCoraPipe* decreases the annotation time by 40% in semantic role labelling, by 60% in named entity annotation, and by 25% in co-reference annotation. This improvement is attributed to "a tool that is very user-oriented, focusing on usability and operational simplicity" (Bertran *et al.*, 2008). However, of the 26 tools mentioned in Table 10, only Bertran *et al.* (2008) specify any improvement in annotation time, but do not provide any details of how the experiment was performed. They also do not provide any detailed statistics about improvement in accuracy. Some of the other software listed in Table 10 mentions improvement but does not specify any detail. For example: "We observed significant acceleration both in correcting the existing treebank and developing new datasets" (Eryigit, 2007); or "… proved to be highly modifiable in response to the evolving task definition and increasing demands for annotation speed and accuracy" (Maeda *et al.*, 2006). It is assumed that the additional time and cost of developing tailor-made software can be justified by the improvement of annotation time and/or accuracy, but none of the listed software mentions this aspect explicitly.

In this chapter we will attempt to determine the effect (in terms of annotation time and accuracy) of different software environments in linguistic data annotation. If it is beneficial to use tailor-made software, we will also investigate whether the benefit can justify the development cost. It is important to note that this chapter does not attempt to investigate the benefits of methods such as bootstrapping or active learning on data annotation, which are used to improve or reduce the data to be annotated. Software that use bootstrapping (i.e. implement the same technology that the project aim to develop) to provide the annotator with pre-annotated data have been proven to be beneficial. Examples of such software include *SemTag* (Dill *et al.*, 2003), *AeroDAML* (Kogut & Holmes, 2001), *BRAT* (Stenetorp *et al.*, 2012), and *JANE* (Tomanek *et al.*, 2007). Software that include these methods are becoming more popular as the technologies exist for mainstream languages. For resource-scarce languages, however, these technologies do not readily exist, and as such are explicitly excluded in the software used in these experiments and subsequent discussions. The focus of this chapter is to investigate different software environments and the effect the environments have on the user's ability to perform a task more efficiently.

3.3 Research questions

Almost no proof exist that it is beneficial to annotate in tailor-made software. Nonetheless, many annotation projects develop their own software in the hope of improving on annotation cost. However, development cost of tailor-made software is often not accounted for when researchers report on the success of their tailor-made software.

When considering whether or not to develop tailor-made software, it is prudent to ask:

- Is it beneficial in terms of time and quality to use tailor-made software instead of domain-specific or general-purpose software?
- If it is beneficial to use tailor-made software, can the additional development cost be justified by the savings in annotation cost?

This chapter will attempt to answer these questions by investigating two tasks performed in different software environments, by annotators of the same level of expertise.

3.4 Experimental setup

3.4.1 Description of tasks

Respondents completed the same two tasks as in Chapter 2 (see 2.4.1), viz. **lemmatisation of text data** (Task A) and **orthographic transcription of audio data** (Task B). Detailed descriptions of the tasks as well as the protocols used in the experiments are provided in Annexure A and Annexure B.

3.4.2 Data

The same data as described in Chapter 2 was used for this experiment (see 2.4.2).

3.4.3 Software environments

For Task A, *CrowdFlower* (Annexure C.1) and Microsoft *Excel* (Annexure C.2) were used as general-purpose software, *LARALite* (Annexure C.3.1) as domain-specific software and *LARAFull* (Annexure C.3.2) as tailor-made software. For Task B, *CrowdFlower* was used as general-purpose software, *Praat*[38] (Annexure C.4) as domain-specific software and *TARA* (Annexure C.5) as tailor-made software.

[38] Although various other domain-specific software transcription environments are available, *Praat* has an extensive user base and is often preferred due to its mature feature set, stability and availability as Open Source software.

For both tasks, certain variables that could influence the results of the experiments were kept constant in order to ensure a controlled experiment:

1. **Hardware:** All experiments were performed on desktop PCs with identical specifications. The experiments were conducted in a student computer laboratory at North-West University. For Task B, all respondents also used identical headphones to ensure similar noise levels and clarity of the audio data.
2. **Presentation of data:** For Task A, the words to be lemmatised were presented as a tokenised list (one token per line) in all four software environments. In *CrowdFlower*, the tokens were provided with an empty text box directly underneath the token where the respondent had to type the lemma, while in *Excel, LARALite* and *LARAFull*, the tokens were provided in the first column and the respondent had to provide the lemma in the second column. For Task B, the recording was divided on sentence level, and each sentence was provided separately to the respondent in all three software environments. He/she had to provide the transcription for each sentence.

3.4.4 Respondents

The criteria for the respondents in this chapter were the same criteria as for novices in Chapter 2 (see 2.4.4). They had the same level of expertise (i.e. undergraduate language students), and were native speakers of the task language (i.e. Afrikaans). All respondents in the experiments participated on a voluntary basis and received a small honorarium for the successful completion of a task. Ten respondents were used in each software environment, for each task, i.e. forty respondents for lemmatisation and thirty respondents for audio transcription. Respondents that participated in the experiments in Chapter 2 were not allowed to partake in these experiments.

3.4.4.1 Training

As reported in Chapter 2 (see 2.4.4.1) the respondents received no training; but all respondents reported on in this chapter received training in the specific software they used for the task as well as training in the task itself. Respondents had to follow specific protocols for both tasks that were developed in separate projects and simplified and customised for these experiments. The training was elucidated with some examples that were not included in the dataset. All respondents received one hour of training.

3.4.5 Evaluation criteria

In this experiment we use the same two sets of evaluation criteria as reported in the previous chapter: one for the task of lemmatisation and a separate set for the task of orthographic transcription. For the task of lemmatisation, evaluations were performed on the time needed to complete the task, the overall performance, capitalisation errors made, spelling errors made, and no response provided (see 2.4.5.1). For the task of orthographic transcription of audio data, time needed to complete the task was measured and all errors present in the respondents' transcriptions were manually annotated and classified into separate categories as described in 2.4.5.2.

3.4.6 Hypothesis

In this experiment the hypothesis is that there is a significant improvement in time and accuracy when using tailor-made software for data annotation. The experiments investigate the hypothesis to determine the viability of this assumption.

3.5 Results, analysis and interpretation

Analysis of the results was performed using two statistical models[39]:

1. **ANOVA tests** were used to compare the average scores of different groups on a dependent variable. ANOVA relies on assumptions of normality of the data and homogeneity of variances.
2. **Kruskal-Wallis tests** (non-parametric tests which are more robust for violation of assumptions than ANOVA) were performed. The Kruskal-Wallis test does not assume a normal population and the null hypothesis is that the populations from which the samples originate have the same median.

For these two tests, a *p*-value smaller than 0.05 is considered sufficient evidence that the result is statistically significant. As in Chapter 2, effect sizes are also reported (see 2.5). Cohen's *d*-value (Cohen, 1988) was used as a measure of practical significance. For the interpretation of the effect size, *d* = ±0.2 is considered a small effect (no practically significant difference), *d* = ±0.5 is considered a medium effect (practically visible difference) and *d* = ±0.8 is considered a large effect (practically significant difference).

[39] We used different models as in Chapter 2, since the sample sizes were balanced in this case.

3.5.1 Task A: Lemmatisation of Afrikaans text data

3.5.1.1 Results

Figure 9 illustrates the average annotation time per respondent in all four software environments. Respondents using *LARAFull* completed the task 49.66% (32.65 minutes) faster than respondents using *CrowdFlower*; 52.67% (36.82 minutes) faster than respondents using *Excel*; and 46.54% (28.81 minutes) faster than respondents using *LARALite*. The main features incorporated in *LARAFull* which may contribute to this saving are the "Apply to all" and the "Same as token" features described in Annexure C.3.

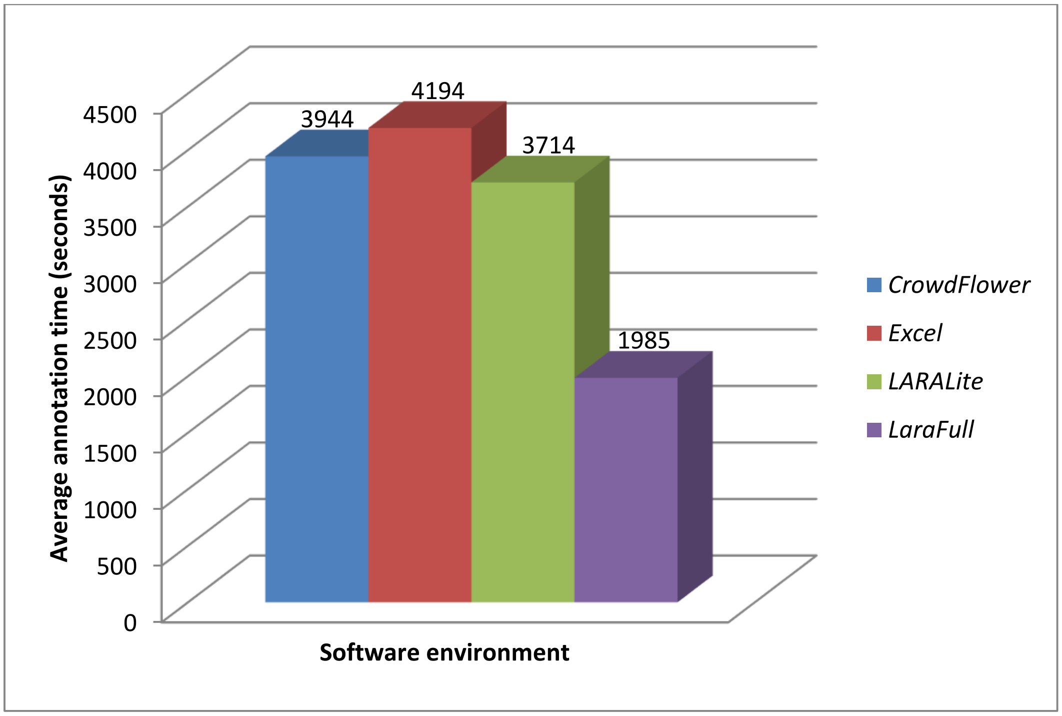


Figure 9: Total annotation time in seconds per environment

To evaluate the overall performance achieved by the respondents, the accuracy, precision, recall and *f*-scores (see Table 11) achieved in the four software environments were compared. The difference in results achieved in each of the four software environments can be attributed to the number of capitalisation errors, spelling errors and instances where the respondent failed to provide an answer, as well as the incorrect lemmatisation of words. Figure 10 shows the total capitalisation errors, spelling errors and empty responses in the four software environments.

	Accuracy	Precision	Recall	*F*-score
CrowdFlower	90.33	93.96	94.43	94.16
Excel	88.63	92.41	92.06	92.22
LARALite	90.44	94.67	93.5	94.07
LARAFull	93.92	97.06	96.68	96.86

Table 11: Accuracy, precision, recall and *f*-score

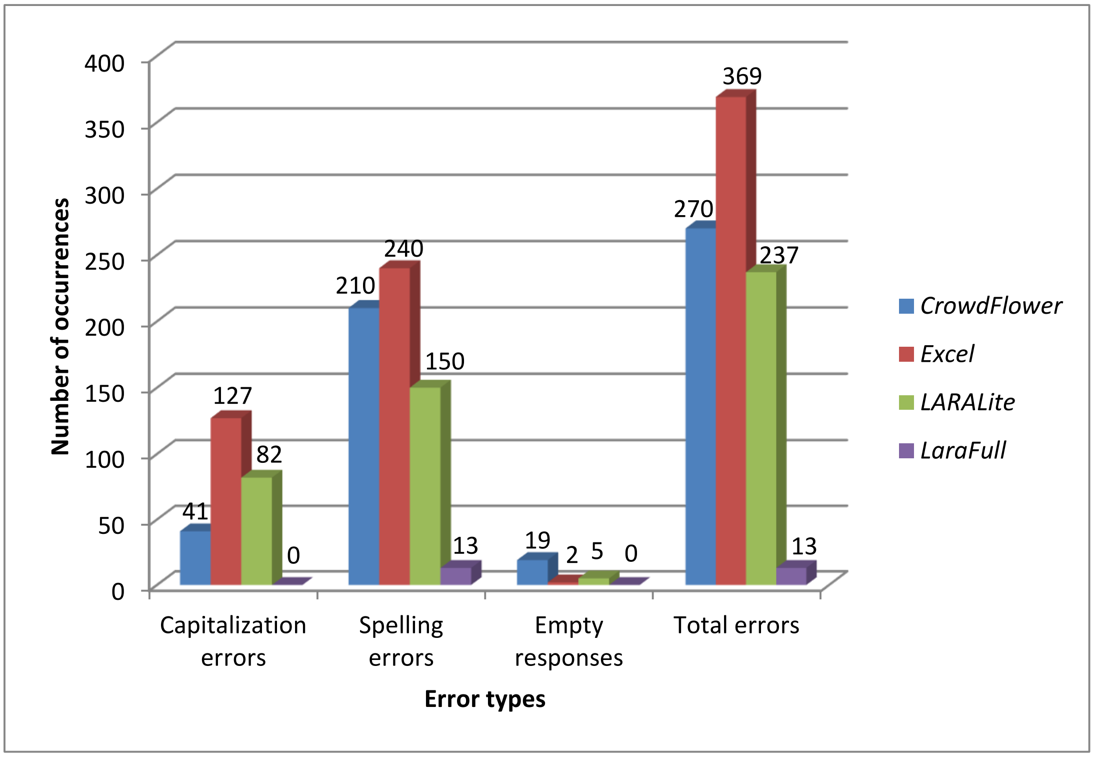


Figure 10: Capitalisation errors, spelling errors and empty responses

Most notable in these results is that no capitalisation errors or empty responses are present in the data annotated in *LARAFull*. This can be attributed to two features in *LARAFull*, i.e. capitalised lemmas are flagged as capitalisation errors if the lemma appears in the lower-case spelling checker lexicon, and empty lemma fields are flagged if a user skips a required entry. The spelling errors present in the data annotated in *LARAFull* are far fewer than in the other three software environments and can be attributed to the feature included in *LARAFull* that automatically flags spelling errors made by the respondent and provides suggestions in a pop-up window. Spelling errors were still present in the data annotated in *LARAFull* as spelling errors were only flagged and the user was able to continue without correcting the error.

3.5.1.2 Statistical analysis and interpretation

Although the results in the previous section show that the overall performance of data annotated in *LARAFull* was better than in the other software environments, and that the data annotated in *LARAFull* contained fewer capitalisation errors, spelling errors and empty responses – further investigations were conducted to determine the significance of these differences. Table 12 shows the mean difference standard deviation, 95% confidence intervals and *p*-values of the ANOVA test.

	CrowdFlower		***Excel***		***LARALite***		***LARAFull***		***P*-value**
	Mean	**Standard deviation**	**Mean**	**Standard deviation**	**Mean**	**Standard deviation**	**Mean**	**Standard deviation**	**Repeated measures ANOVA**
	95% Confidence interval		**95% Confidence interval**		**95% Confidence interval**		**95% Confidence interval**		
	Lower	**Upper**	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	**Upper**	
Time (s)	3943.900	736.355	4194.300	983.927	3713.500	527.753	1985.200	260.496	< 0.001
	3417.143	4470.657	3490.441	4898.159	3335.968	4091.032	1798.853	2171.547	
Accuracy	0.903	0.030	0.886	0.050	0.904	0.040	0.939	0.018	0.021
	0.882	0.925	0.850	0.922	0.875	0.933	0.926	0.952	
Precision	0.691	0.119	0.655	0.109	0.668	0.114	0.758	0.106	0.190
	0.605	0.776	0.577	0.733	0.587	0.750	0.683	0.834	
Recall	0.641	0.126	0.667	0.129	0.708	0.081	0.762	0.040	0.056
	0.550	0.731	0.575	0.759	0.650	0.766	0.733	0.791	
F-score	0.656	0.095	0.656	0.111	0.684	0.087	0.756	0.053	0.053
	0.588	0.723	0.577	0.735	0.622	0.747	0.718	0.794	
Capitalisation errors	4.100	6.027	12.700	26.264	8.200	16.599	0.000	0.000	0.329
	-0.211	8.411	-6.088	31.488	-3.674	20.074	0.000	0.000	
Spelling errors	21.000	14.952	24.000	14.832	15.000	12.614	1.300	2.263	0.001
	10.304	31.696	13.390	34.610	5.977	24.023	-0.319	2.919	
Empty responses	1.900	1.912	0.200	0.422	0.500	0.972	0.000	0.000	0.002
	0.532	3.268	-0.102	0.502	-0.195	1.195	0.000	0.000	

Table 12: Means, standard deviation and *p*-values for Task A

Statistically significant differences were present on time ($p < 0.001$), accuracy ($p = 0.021$), spelling errors ($p = 0.001$) and empty responses ($p = 0.002$) of data annotated in the different software environments. The pairwise comparisons between the different software environments (see Table 13) showed statistically significant differences between *CrowdFlower* and *Excel* on empty responses ($p = 0.007$), *CrowdFlower* and *LARALite* on empty responses ($p = 0.034$) and *CrowdFlower* and *LARAFull* on time

($p < 0.001$), recall ($p = 0.050$), spelling errors ($p = 0.006$) and empty responses ($p = 0.002$). No statistically significant differences were present between *Excel* and *LARALite*, while statistically significant differences were present on time ($p < 0.001$), accuracy ($p = 0.014$) and spelling errors ($p = 0.001$) between *Excel* and *LARAFull*. Statistical differences between *LARALite* and *LARAFull* were only present on time ($p < 0.001$).

Practically significant differences (see Table 13) were present between *CrowdFlower* and *Excel* as well as between *CrowdFlower* and *LARALite* on empty responses. Practically significant differences were present on all variables except precision between *CrowdFlower* and *LARAFull*. No practically significant differences were present between *Excel* and *LARALite*. The comparison between *LARAFull* and *Excel* as well as between *LARAFull* and *LARALite* showed practically significant differences on all variables except capitalisation errors and empty responses.

	CrowdFlower vs. Excel		***CrowdFlower vs. LARALite***		***CrowdFlower vs. LARAFull***		***Excel vs. LARALite***		***Excel vs. LARAFull***		***LARALite vs. LARAFull***	
	p	***d***	***p***	***d***	***p***	***d***	***p***	***d***	***p***	***d***	***p***	***d***
Time	0.844	-0.304	0.874	0.379	< 0.001	3.738	0.404	0.642	< 0.001	3.235	< 0.001	4.378
Accuracy	0.731	0.435	1.000	-0.030	0.147	-1.534	0.691	-0.419	0.014	-1.487	0.167	-1.189
Precision	0.892	0.333	0.971	0.208	0.539	-0.627	0.993	-0.123	0.186	-1.010	0.294	-0.862
Recall	0.939	-0.215	0.451	-0.667	0.050	-1.364	0.795	-0.401	0.167	-1.049	0.632	-0.891
F-score	1.000	0.000	0.887	-0.324	0.074	-1.370	0.888	-0.296	0.075	-1.212	0.293	-1.054
Capitalisation errors	0.621	-0.476	0.938	-0.346	0.938	1.014	0.920	0.216	0.293	0.721	0.656	0.736
Spelling errors	0.948	-0.212	0.699	0.457	0.006	1.942	0.374	0.689	0.001	2.255	0.080	1.594
Empty responses	0.007	1.294	0.034	0.973	0.002	1.481	0.927	-0.422	0.977	0.706	0.737	0.767

Table 13: Pairwise comparisons and d-values between software used in Task A

The Kruskal-Wallis tests showed statistical significant differences on time, accuracy, *f*-score, capitalisation errors, spelling errors, and empty responses (see Table 14). Further pairwise comparisons showed no statistically significant differences between *CrowdFlower* and *Excel*, *CrowdFlower* and *LARALite* or *Excel* and *LARALite*. Statistically significant differences were present between *CrowdFlower* and *LARAFull* on time, spelling errors and empty responses, between *Excel* and *LARAFull* on time, accuracy, *f*-score and spelling errors and between *LARALite* and *LARAFull* on time and spelling errors.

	Kruskal-Wallis	***CrowdFlower* vs. *Excel***	***CrowdFlower* vs. *LARALite***	***CrowdFlower* vs. *LARAFull***	***Excel* vs. *LARALite***	***Excel* vs. *LARAFull***	***LARALite* vs. *LARAFull***
Time	< 0.001	1.000	1.000	0.001	1.000	< 0.001	0.005
Accuracy	0.022	1.000	1.000	0.137	1.000	0.019	0.244
Precision	0.155	1.000	1.000	0.976	1.000	0.268	0.306
Recall	0.051	1.000	1.000	0.056	1.000	0.184	0.785
F-score	0.044	1.000	1.000	0.096	1.000	0.077	0.335
Capitalisation errors	0.020	1.000	1.000	0.066	1.000	0.636	0.115
Spelling errors	< 0.001	1.000	1.000	0.002	1.000	< 0.001	0.029
Empty responses	0.003	0.167	0.471	0.026	1.000	1.000	1.000

Table 14: *P*-values of Kruskal-Wallis tests and pairwise comparisons for Task A

These results show a statistically significant benefit of using *LARAFull* over *CrowdFlower*, *Excel* and *LARALite*, especially in terms of time needed to complete the task and to a lesser extent on spelling errors. Practically significant differences were also present on the majority of variables when comparing *LARAFull* with the other software environments. Based on these results and analysis, it seems that for the task of lemmatisation of Afrikaans data, it is beneficial to use tailor-made software instead of general-purpose or domain-specific software.

3.5.2 Task B: Orthographic transcription of Afrikaans audio data

3.5.2.1 Results

Figure 11 illustrates the difference in the average time per respondent needed to transcribe the audio data in the three software environments. The total improvement of time needed to perform the transcriptions in *TARA* was 939 seconds (15.65 minutes) compared to *CrowdFlower* and 600 seconds (10 minutes) compared to *Praat.*

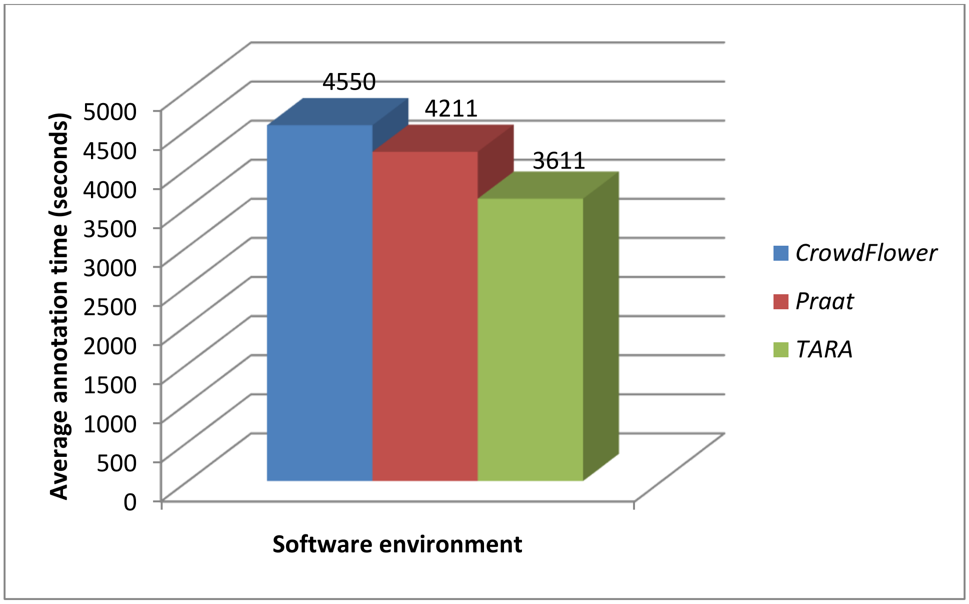


Figure 11: Average transcription time in each software environment

Figure 12 illustrates errors made by respondents in each of the three software environments. Table 15, Table 16 and Table 17 provide a breakdown of transcription, language and protocol errors.

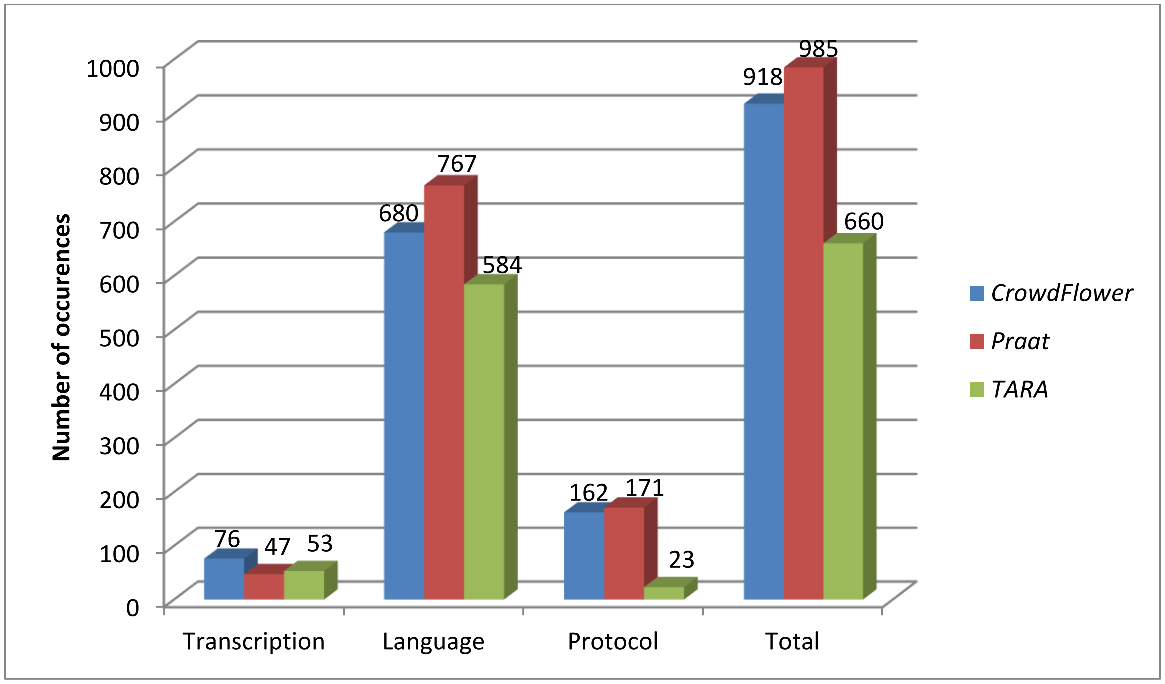


Figure 12: Total annotated errors made in each software environment

From the breakdown of transcription errors, it is evident that none of the software environments influence the number of transcription errors made by the respondents. It is interesting to note that *TARA* did not include any features which might be beneficial in reducing transcription errors (only spelling errors and protocol errors). Features aimed at reducing spelling errors and protocol errors were beneficial in reducing the errors made by the respondents (see Table 16 and Table 17). *TARA* included a spelling checker, and as can be seen in the breakdown of language errors, this feature was beneficial and reduced the number of spelling mistakes in the data transcribed in *TARA* when compared to the data transcribed in *CrowdFlower* and *Praat*.

	Insertions	**Deletions**	**Substitutions**	**Transpositions**
CrowdFlower	3	29	44	0
Praat	7	6	34	0
TARA	1	14	38	0

Table 15: Breakdown of transcription errors

	Spelling error	**Capitalisation**	**Punctuation**	**Hyphen**	**Compound**
CrowdFlower	219	131	179	74	77
Praat	234	132	223	87	91
TARA	126	127	180	70	81

Table 16: Breakdown of language errors

	Capitalisation	**Number**	**Abbreviation**	**Acronym**	**Punctuation**	**White Space**	**Terminator**
CrowdFlower	16	5	8	9	36	34	54
Praat	14	4	3	5	27	70	48
TARA	8	0	3	10	1	1	0

Table 17: Breakdown of protocol errors

Most notable is the reduction of protocol errors in the data transcribed in *TARA* when compared to data transcribed in *CrowdFlower* and *Praat*. *TARA* included features to flag incorrect capitalisation of words, digits, invalid punctuation, excessive white space and invalid terminators. The total number of errors made by respondents in *TARA* (660 errors) is noticeably less than the errors made by respondents in *CrowdFlower* (918 errors) and in *Praat* (985 errors). This indicates that is beneficial to include features that automatically flag errors.

3.5.2.2 Statistical analysis and interpretation

Although the results in the previous section show that the overall performance in *TARA* is better than in the other software environments and that the data annotated in *TARA* contains fewer language and protocol errors, further investigations were conducted to determine the significance of these differences. Table 18 shows the mean difference, standard deviation, 95% confidence intervals and *p*-values of the ANOVA test.

	CrowdFlower		***Praat***		***TARA***		***P*-value**
	Mean	**Standard deviation**	**Mean**	**Standard deviation**	**Mean**	**Standard deviation**	**Repeated measures ANOVA**
Time (s)	4550.100	694.837	4210.500	623.537	3610.700	625.521	0.011
Total Errors	91.800	25.148	98.500	42.513	66.000	16.931	0.055
Transcription errors							
Insertions	0.300	0.483	0.700	1.160	0.100	0.316	0.207
Deletions	2.900	2.558	0.600	0.699	1.400	1.897	0.034
Substitutions	4.400	2.836	3.400	3.307	3.800	2.150	0.727
Transpositions	0.000	0.000	0.000	0.000	0.000	0.000	n/a
Language errors							
Spelling error	21.900	10.744	23.400	13.066	12.600	3.658	0.047
Capitalisation	13.100	5.744	13.200	5.203	12.700	5.982	0.978
Punctuation	17.900	6.244	22.300	11.096	18.000	7.257	0.426
Hyphen	7.400	5.147	8.700	4.448	7.000	5.249	0.728
Compound	7.700	4.398	9.100	6.839	8.100	4.557	0.837
Protocol errors							
Capitalisation	1.600	1.350	1.400	1.075	0.800	0.632	0.233
Number	0.500	1.269	0.400	1.265	0.000	0.000	0.528
Abbreviation	0.800	1.619	0.300	0.483	0.300	0.949	0.522
Acronym	0.900	0.876	0.500	0.707	1.000	0.816	0.352
Punctuation	3.600	3.307	2.700	6.499	0.100	0.316	0.175
White space	3.400	2.875	7.000	12.481	0.100	0.316	0.133
Terminator	5.400	8.733	4.800	8.483	0.000	0.000	0.189

Table 18: Mean values, standard deviation and *p*-values of variables in Task B

The ANOVA *p*-values illustrate that statistically significant differences were present on time ($p < 0.011$), deletions ($p = 0.034$) and spelling errors ($p = 0.047$). As with the analysis of Task A, pairwise comparisons were conducted to determine where the differences lay. The pairwise comparisons (see Table 19) showed statistically significant differences between *CrowdFlower* and *Praat* on deletions ($p = 0.029$), between *CrowdFlower* and *TARA* on time ($p = 0.009$) and between *Praat* and *TARA* on punctuation (language) ($p < 0.001$) and capitalisation (protocol) ($p < 0.001$). Practically significant differences were present between *CrowdFlower* and *Praat* on deletions; between *CrowdFlower* and *TARA* on time, total errors, spelling errors, punctuation (protocol) and white space; and between *Praat* and *TARA* on time and spelling errors (see Table 19).

	CrowdFlower* vs. *Praat		***CrowdFlower* vs. *TARA***		***Praat* vs. *TARA***	
	p	***d***	***p***	***d***	***p***	***d***
Time (s)	0.481	0.489	0.009	1.352	0.116	0.959
Total Errors	0.873	-0.158	0.154	1.026	0.058	0.764
Transcription errors						
Insertions	0.466	-0.345	0.822	0.414	0.191	0.517
Deletions	0.029	0.899	0.195	0.586	0.614	-0.422
Substitutions	0.452	0.434	0.957	0.134	0.624	-0.333
Transpositions	n/a	n/a	n/a	n/a	n/a	n/a
Language errors						
Spelling error	0.940	-0.115	0.113	0.866	0.057	0.827
Capitalisation	0.999	-0.017	0.986	0.067	0.979	0.084
Punctuation	0.485	-0.397	1.000	-0.014	< 0.001	0.388
Hyphen	0.829	-0.253	0.982	0.076	0.727	0.324
Compound	0.831	-0.205	0.985	-0.088	0.910	0.146
Protocol errors						
Capitalisation	0.907	0.148	0.229	0.593	< 0.001	0.558
Number	0.975	0.079	0.534	0.394	0.667	0.316
Abbreviation	0.584	0.309	0.584	0.309	1.000	0.000
Acronym	0.514	0.457	0.958	-0.114	0.359	-0.612
Punctuation	0.882	0.138	0.171	1.059	0.365	0.400
White space	0.529	-0.288	0.585	1.148	0.112	0.553
Terminator	0.980	0.069	0.217	0.618	0.295	0.566

Table 19: Pairwise comparisons and effect sizes of Task B

The Kruskal-Wallis tests (see Table 20) also show statistical significant differences on time ($p = 0.018$), spelling errors ($p = 0.044$), punctuation (protocol) ($p = 0.011$), white space ($p = 0.001$) and terminator (*p*

= 0.003). The pairwise comparisons showed no statistical significant differences between *CrowdFlower* and *Praat*. Statistically significant differences were present between *CrowdFlower* and *TARA* on time (*p* = 0.016), punctuation (protocol) (*p* = 0.020), white space (*p* = 0.003), and terminator (*p* = 0.008). A statistically significant difference was present between *Praat* and *TARA* on white space (*p* = 0.004).

	Kruskal-Wallis	***CrowdFlower* vs. *Praat***	***CrowdFlower* vs. *TARA***	***Praat* vs. *TARA***
Time (s)	0.018	1.000	0.016	0.191
Total Errors	0.053	1.000	0.112	0.105
Transcription errors				
Insertions	0.414	1.000	1.000	1.000
Deletions	0.061	0.076	0.432	1.000
Substitutions	0.447	0.640	1.000	1.000
Transpositions	1.000	1.000	1.000	1.000
Language errors				
Spelling error	0.044	1.000	0.074	0.119
Capitalisation	0.979	1.000	1.000	1.000
Punctuation	0.659	1.000	1.000	1.000
Hyphen	0.322	0.599	1.000	0.573
Compound	0.996	1.000	1.000	1.000
Protocol errors				
Capitalisation	0.309	1.000	0.560	0.743
Number	0.355	1.000	1.000	1.000
Abbreviation	0.548	1.000	1.000	1.000
Acronym	0.338	0.929	1.000	0.573
Punctuation	0.011	0.599	0.020	0.454
White space	0.001	1.000	0.003	0.004
Terminator	0.003	1.000	0.008	0.060

Table 20: Kruskal-Wallis test and pairwise comparisons

These results show a statistically significant benefit of using *TARA*, compared to *CrowdFlower* and *Praat,* especially in terms of time needed to complete the task and, to a lesser extent, in terms of spelling errors. Based on these results and analysis, it seems that for the task of orthographic transcription of Afrikaans audio data, it is beneficial to use tailor-made software instead of general-purpose or domain-specific software.

3.6 Development time vs. benefit

The aim of this section is to determine if the savings in annotation time and/or accuracy can justify the additional development time. For illustrative purposes we apply the possible saving to two projects funded by the South African Department of Arts and Culture and compare the possible savings to development time. In both comparisons, we determine the worst case scenario, where we assume that the respondents will not improve on speed or accuracy even though annotators usually become more sufficient in the task. By assuming that the annotators continue at the same speed, the saving in annotation time is calculated at the minimum, and as such, if a project scope falls within the minimum required dataset size, their choice of software can be based on these findings with confidence. In reality, the benefits will be to a larger extent[40].

The hourly rate of annotators can vary significantly; for example, a novice will probably work at a much lower rate than an expert. The rates of experts also differ, and in our previous experience, especially for languages with very few linguistic experts, their rates are in most cases higher than that of system developers. Due to this variance in hourly costs, we assume that the hourly development cost is equal to the hourly annotation cost, and as such only compare development time to annotation time. Projects that consider developing tailor-made software, should calculate the exact cost implication using actual and up to date hourly rates.

The project applicable to the task of lemmatisation aimed to annotate 50,000 tokens for ten of the official South African languages (English was excluded), on four levels – tokenisation, lemmatisation, part-of-speech tagging and morphological analysis. Thus, a total of 500,000 tokens were to be lemmatised. *LARAFull* showed a saving of 28 minutes per 1,000 tokens (see 3.5.1.1), implying a saving of about 233 hours in terms of time to perform lemmatisation of 500,000 tokens. If the same savings are assumed on the tokenisation, part-of-speech tagging and morphological analysis (i.e. two million tokens in total for all four tasks), a total saving of 932 hours could be feasible. *LARA2* (see Annexure C.3 for a detailed description) was developed as a single environment in which all four levels of annotations, of all

[40] In a related project, we further examined the impact on saving in terms of time in *LARAFull* and used four trained, experienced annotators to lemmatise three sets of five thousand words per set. Their average annotation times reduced with 28.15% from the first to the second set and with 41.02% from the first to the third set. The reduction from the second to the third set was less prominent with a 17.91% reduction. This reduction in annotation time was specifically attributed to the “Apply to all” feature, as less unseen words are present in the new datasets (i.e. words in following sets have already been lemmatised in previous sets), as well as to the annotators becoming more experienced in the task.

ten languages, could be performed. The development time for *LARA2* was 400 hours, including design, development and testing. When considering the total saving of 932 hours on the annotation of the total of two million tokens, we can calculate that the 400 hours of development time can be justified by the saving in annotation time of roughly 860,000 tokens.

The project applicable to the task of orthographic transcription aimed to transcribe fifty hours speech data for each of the eleven official South African languages; thus 550 hours of audio data needed to be transcribed. If we extrapolate the difference in transcription time between *TARA* and *Praat* (i.e. ten minutes saving on six minutes of audio data; see 3.5.2.1) to fifty hours of audio data, the reduction in transcription time is about 83 hours. This implies that using *TARA* to perform the transcriptions of the 550 hours of audio data in the aforementioned project could save the project 913 hours in transcription time. The development time for *TARA* was also 400 hours including design, development and testing. From this we can estimate that the development cost can be justified by the saving in annotation time of 240 hours of audio data.

If we make a conservative generalisation from these estimates, it would suggest that for the task of text-based annotation, the development time of tailor-made software can only be justified if more than one million tokens are to be annotated. For the task of audio-based annotation, this can be estimated at 300 hours of audio data. The conclusion can therefore be made that the savings in annotation time can only justify the additional development time and cost, if the scope of the project is sufficient (i.e. if one million tokens are to be annotated, or if 300 hours of audio data is to be transcribed). As HLT-related projects for resource-scarce languages generally do not entail projects of these sizes, it would seem that if we only take development time vs. annotation time into account, the development of tailor-made software is not necessarily an efficient method for the creation of resources for resource-scarce languages. This aspect will be further discussed in Chapter 5.

Apart from the development time vs. annotation time, one should also consider the increase in annotation quality. In the task of lemmatisation, statistically significantly fewer errors were present in the data annotated in the tailor-made software, specifically capitalisation and spelling errors. For the task of audio transcription, all transcriptions performed in the tailor-made software contained no fatal errors (invalid punctuation, white space, noise markers, numbers and terminator errors[41]) as the software automatically checked for these errors and did not allow a user to continue until the error was

[41] These errors are referred to as fatal errors as they cannot be valid in any context, as opposed to, for example, capitalisation at the start of a sentence that might be valid if the first word is a name.

fixed. In contrast, transcriptions performed in the domain-specific software contained 145 fatal errors. The reduction in annotation errors will result in a saving of time needed for quality control, and if quality is of major concern to a project, the benefit of using tailor-made software should also be taken into account.

One approach that projects could take is rather to customise existing domain-specific software by adding specific features aimed at reducing the annotation time and increasing the annotation quality. This would reduce the development time, but still include the benefits of using tailor-made software.

3.7 Conclusion

The first aim of this chapter was to establish the benefits in terms of time and quality when using tailor-made software instead of domain-specific or general-purpose software. The second aim was to establish if additional development time of tailor-made software could be justified by the savings in annotation time.

Firstly, we provided an overview of the main differences between general-purpose software, domain-specific software and tailor-made software. Next, we discussed studies involving various software environments developed for the purpose of annotation, and showed that a major shortcoming of these studies was that no evidence was provided to support the claims of improvement in terms of annotation time or accuracy. None of the studies compared development time with savings in annotation time, and as such, no conclusion could be made regarding whether additional development time could be justified by a saving in annotation time.

Next, we conducted systematic experiments in different software environments, while using respondents of the same skill level. The respondents also received the same extent of training and used identical hardware, in order to ensure that the only variable in the experiments were the software environments in which the tasks were completed. The respondents completed the same tasks as in the previous chapter, i.e. lemmatisation of 1,000 words (Task A) and the orthographic transcription of six minutes of audio data (Task B).

For the task of lemmatisation, forty respondents completed the task in four software environments (ten per environment), and the results showed that the respondents who used the tailor-made software (i.e. *LARAFull*) completed the task between 28 and 36 minutes faster (statistically significant) than respondents who used the general-purpose and the domain-specific software. Data annotated in the tailor-made software also contained statistically significantly fewer capitalisation and spelling errors than the data annotated using the other software environments.

For the task of orthographic transcription of audio data, thirty respondents completed the task in three software environments (ten per environment). The results showed that the respondents who used the tailor-made software (i.e. *TARA*) also completed the task in a statistically significantly shorter time than the respondents using the other software environments. The data annotated in the tailor-made software also contained fewer errors (statistically significant) than the data annotated in the other software environments, especially with regard to spelling errors and protocol errors.

From these results, we concluded that is beneficial to use tailor-made software instead of general-purpose or domain-specific software. In order to establish whether the additional development time of tailor-made software could be justified by the savings in annotation time, we extrapolated the results of Task A and B to real world annotation projects that included the annotation of two million tokens and the transcription of 550 hours of audio data. This comparison of annotation time with development time showed that, for these two specific projects, the benefit of the saving of annotation time justifies the development time. However, if we generalise the saving of annotation time, we conclude that the scope of a project has to be in excess of one million tokens or 300 hours of audio data for development time to be justified. As resource-scarce languages rarely conduct projects to this extent, it seems as if the development of tailor-made software is not a viable option for the creation of resources for resource-scarce languages.

The comparison of development time and annotation time does not, however, take the increase of quality and subsequent reduction in quality control, into account. Our conclusion and subsequent recommendation on whether or not to develop tailor-made software is that projects should rather customise an existing domain-specific software environment by adding specific beneficial features in order to reduce development time and still include the benefits of a tailor-made software environment.

4 Chapter 4: The effect of data quality vs. data quantity

4.1 Introduction

The previous two chapters focused on factors related to the annotator (i.e. skill level) and his/her environment (i.e. the user interface that the annotator uses). In this chapter, the focus is on the practical implications of using the annotator to either increase the quality or the quantity of annotated data to be used as training data for the development of HLT systems. Given the limited financial resources available for resource-scarce languages, projects often have to decide whether quality control needs to be performed or if they should rather annotate more data. Because both data quality and quantity have an influence on the performance of the resulting system, the aim of this chapter is to establish which of the two is the most beneficial to a system.

The following section provides an overview of the extent of quality control implemented by various HLT projects and provides a motivation for conducting experiments to establish whether it is more beneficial to focus on the quality or quantity of training data. Section 4.4 describes the tasks, data and evaluation criteria used in the experiments and section 4.5 provides the results, analysis and interpretation.

4.2 Literature survey

Over the past twenty years, supervised learning through methods such as machine learning, has become one of the dominant approaches in the development of HLTs, as stated by, amongst others, Cardie (2005), Chattopadhyay (2013) and Wang and Li (2013). Instead of using rule-based methods that require deep linguistic knowledge and/or specialised knowledge, time and money is spent on developing re-usable, annotated data. For mainstream languages such as English, high quality training data is readily available and projects often focus on improving machine learning-based classifiers through feature optimisation, algorithm improvement, data selection, etc. However, for resource-scarce languages, the lack of sufficient training data is a major concern (Davel *et al.*, 2011; Denis & Sagot, 2009).

Projects that aim to develop HLTs for resource-scarce languages through machine learning are often faced with two options to improve the accuracy of classifiers: either to increase the quality of the training data[42], or to increase the quantity. Machine learning methods are dependent on training data (i.e. a model is trained on the data provided for the system), and if more training data is provided for a

[42] Training data often contain errors due to insufficient quality control, using multiple annotators, bias by annotators, time and budget constraints, etc.

machine learning system it can learn more, resulting in a more accurate classifier. Similar to the benefit of more data, training data of higher quality also results in a system that can better generalise and predict instances which are not in the training data, thus also improving the accuracy of the classifier. Although increasing the quality or the quantity of training data will benefit a classifier, both require additional effort in terms of time and associated cost. Because of budget constraints, projects often have to choose where to invest, but it is often not clear which will be more beneficial to a classifier.

Projects (e.g. Aduriz *et al.*, 2003; Bada *et al.*, 2012; Dang *et al.*, 2002; Xue & Zhou, 2010; Zaghouani *et al.*, 2010) often assume that error-free training data is an essential part of the training of machine learning systems in order for the systems to achieve the best possible performance, and it is generally assumed that errors in training data will be detrimental for the machine learning algorithm (Hall & Smith, 1998; Hand, 2007; Sheng *et al.*, 2008). In many HLT-related annotation projects, data is manually annotated and might contain some degree of "noise", or annotation errors (Hand, 2007). Thus, the quality of annotated data is a major concern for such projects, and effort (in terms of time and associated cost) is invested to ensure that the annotations are as error-free as possible.

Errors in manually annotated data can be the result of errors in perception (comprehension errors or miss-hearing), the skill level of the annotator (an expert annotator versus a novice annotator), accidental errors (e.g. typing errors), language errors, errors due to inadequate training, unclear or incomplete protocols, etc. As a countermeasure for these errors, and in order to create data that is as error-free as possible, many annotation projects (e.g. Corston-Oliver & Dolan, 1999; Dang *et al.*, 2002; Maamouri & Bies, 2004; Min, 2013; Palmer *et al.*, 2005a; Palmer *et al.*, 2005b) implement various levels of quality control to identify and correct annotation errors in training data. These levels range from automatic identification of errors (e.g. using a spelling checker to verify the correct spelling of words (Oosthuizen *et al.*, 2010), performing quality control on a random or automatically identified subset of data (Sheng *et al.*, 2008), to annotating the complete dataset by numerous people (Maamouri & Bies, 2004; Min, 2013; Rose *et al.*, 2002). When performing multiple annotations, each instance of the data is annotated twice – or in some cases even more often (Maamouri & Bies, 2004). Disagreements are resolved either by using a third annotator (often an expert in the field) to decide on the correct instance, or by discarding the disagreements. If more than two annotators are used, some form of voting would be utilised. This method of using multiple annotators to annotate the same data is referred to as double-blind annotation (Clark *et al.*, 2010).

By using methods such as double-blind annotation, data that is less error prone than single-base annotated (i.e. a particular set of data is only annotated by one annotator, even if multiple annotators are used in the project) data is created, and is often referred to as a gold standard[43]. Some large-scale projects that use double-blind annotation include *OntoNotes* (Min, 2013), which aim to annotate over a million words each of English and Chinese, and half-a-million words of Arabic. Annotation is performed on different layers that include structural information (syntax and predicate argument structure) and shallow semantics (word sense linked to an ontology, and co-reference). Quality of the annotations is measured in terms of inter-annotator agreement and each layer of annotation aims to achieve at least 90% agreement. Other projects such as *PropBank* (Palmer *et al.*, 2005b) and the *Penn Chinese Treebank* (Palmer *et al.*, 2005a) also use double-blind annotation.

Although double-blind annotation is not frequently used in the creation of orthographically transcribed audio data, specifically for use as training data for ASR systems, transcription projects do include quality control. Some examples of quality control in transcription projects include the spoken Dutch corpus (Boves & Oostdijk, 2003), where transcriptions of one transcriber are checked by another transcriber; the AST corpus (Roux *et al.*, 2004), where transcriptions were imported into a single document and proof-read for specification and spelling mistakes; *SpeechOcean*[44] that provides ASR corpora for more than forty languages and perform various levels of quality control to ensure 98%-99.5% transcription accuracy, etc. In reference to resource-scarce languages, and specifically the South African context, De Wet *et al.* (2006) make a recommendation that: “the transcribers’ work should be submitted to quality checks at regular intervals”. Although performing quality control does not imply that double the time is spent on transcriptions, as is the case when using double-blind transcription, additional quality control does imply that more time is spent to create high quality data. In a crowdsourcing context, however, audio data is usually transcribed more than once. Some examples include Novotney and Callison-Burch (2010) that requested three transcriptions for each utterance, or Akasaka (2009) and Parent and Eskenazi (2010) that requested five.

The creation of a gold standard is a valuable resource that can be used as the basis for further annotations or as a test set on which to evaluate different systems, but if the aim of a project is to develop a system of adequate performance, additional measures, such as double-blind annotations or

[43] The term “gold standard” is also used in other contexts as a baseline, but in this study gold standard data is interpreted as a dataset of (near) perfect quality.

[44] http://www.speechocean.com/en-ASR-Corpora/Index.html

extensive quality control, to improve the quality might not be necessary. It is commonly accepted that higher quality annotated data will result in a higher quality system, but it is not necessarily so that the percentage of errors in annotated data will have a linear effect on the performance of the system. For example, if 5% errors are present in the training data, this will not necessarily result in a system with 5% lower performance. It can also not be assumed that the same degree of noise in training data of different quantities will have the same influence on a system's performance. Most machine learning algorithms are designed to be resilient to the presence of errors (or noise) in training data. As such, noise in training data does not necessarily have as great an impact on system performance as is commonly expected. Some studies (see 2.2 for details on these studies) investigated the effect of noise on systems, specifically with regards to data obtained via crowdsourcing, but these studies do not usually include detailed information on the specific level and types of errors in the data. They report disagreement or inter-annotator agreement with reference transcriptions, and the resulting difference in system performance. For example, Novotney and Callison-Burch (2010) compared ASR systems trained with "high quality transcriptions" with systems that had a 23% disagreement with the high quality transcriptions and found, that the WER was only 2.5% worse.

Several studies have been conducted to determine the effect of different types of noise on different machine learning algorithms (Kalapanidas *et al.*, 2003; Khoshgoftaar *et al.*, 2010; Nettleton *et al.*, 2010; Zhu & Wu, 2004), as well as methods to reduce the effect of noise on learning algorithms (Aha, 1992; Rebbapragada & Brodley, 2007; Zhu *et al.*, 2003). These studies show that noise has a varying level of impact on different algorithms, but they also show that learning algorithms are able to accommodate these errors to some extent and still produce systems with acceptable performance. Even though proof exists that machine learning algorithms can accommodate errors in data, a common trend of annotation projects is still to use double-blind annotation for the creation of training data for machine learning systems.

Another factor which is often overlooked is that double the quantity of data (or more if an adjudicator or more than two annotators are used) can be annotated if single-base annotation is used, for the same amount of effort in terms of time needed for the annotation task and the associated cost of the annotations, by using multiple annotators to annotate different datasets. This can also be done by using one annotator to annotate all the data, but by using only one annotator, the risk of introducing annotator bias is increased. The practical effect on system performance of less, "clean" data versus more, erroneous data has not been investigated to the same extent as has the effect of errors on learning algorithms.

In our literature study, we could only find one limited study that explicitly and systematically investigated the effect on system performance of less, "clean" data versus more, erroneous data. Dligach *et al.* (2010) conducted experiments where they compared word sense disambiguation systems that were trained using single-base annotated, double-annotated and adjudicated data as training data. They calculated a cost per training instance for each of the different types of training data and compiled training sets containing different numbers of training instances, but that had the same cost. These training sets were used to train support vector machine (SVM) classifiers and the performance of the systems was compared. They found that the systems trained with single-base annotated data outperformed the other systems if the systems were trained on datasets that had the same cost. In the context of ASR systems, De Wet and De Vries (2013), De Vries *et al.* (2014) and Modipa *et al.* (2013) mentioned the effect of more versus cleaner data, but did not include specific details on the level and types of errors present in the data. Nonetheless, their finding also supports the notion that more date equals better data. The findings of these two studies illustrate a crucial aspect in the training of systems, which is that the quantity of training data has an immense influence on system performance.

Given that both data quality and quantity have an influence on the performance of the resulting system, the aim of this chapter is to investigate the effect of data quality versus data quantity by conducting systematic experiments with training data of varying quality and quantity.

4.3 Research question

Many annotation projects still use double-blind annotation for the development of training data for machine learning systems (e.g. Maamouri & Bies, 2004; Min, 2013; Palmer *et al.*, 2005a; Palmer *et al.*, 2005b; Rose *et al.*, 2002), as it is assumed that errors in the data will have a substantial impact on system performance. Additionally, given the limited financial resources available for resource-scarce languages, projects often have to decide whether quality control needs to be performed or if they should rather annotate more data. Although both training data quality and quantity have an influence on the performance of the resulting system, it is not clear which of these two are the most beneficial to a system. Thus, it is prudent to ask:

- Is it more beneficial to focus on the quality or the quantity of training data?

4.4 Experimental setup

In order to determine if it is more beneficial to focus on the quality or the quantity of training data, two sets of experiments were conducted. In Experiment 1, we compare systems trained with data of the same quantity, but with varying levels of quality, in order to initially establish the effect of errors on system performance. In Experiment 2, we compare systems trained with gold standard data to systems trained with lower quality but double the quantity of data, in order to establish whether increasing the quantity of lower quality training data will affect system performance. We assume that the error correction when combining two sets of data will result in halve the quantity gold standard data. This approach is optimistic, as two sets of data do not usually result in gold standard data without adjudication of the differences, but the bias is in favour of the systems trained with the gold standard data, further ensuring the reliability of the results.

4.4.1 Description of tasks

For both experiments, we conducted two distinct tasks: developing a lemmatiser for Afrikaans (Task A) and developing an ASR system for Afrikaans (Task B).

4.4.1.1 Task A: Developing a lemmatiser for Afrikaans

Groenewald (2006) developed a data-driven lemmatiser (*Lemma Identifiseerder vir Afrikaans* (LIA)) using memory-based learning with the *Tilburg Memory-Based Learner* (*TiMBL*) (Daelemans *et al.*, 2001), and achieved an accuracy of 92.8%. The data developed and used by Groenewald (2006) was used as gold standard data in these experiments. This data consists of 73,620 words, each with a class indicating the affixes to be removed (if the word is not already in the base form) in order to identify the lemma. For example, the class “*Rtjies>*” indicates that “*tjies*” should be removed from the end of the word. Words that already appear in the base form have 0 as a class. A total of 271 classes are present in the data. The data was automatically extracted and classified, and various iterations of quality control were performed in order to produce a gold standard. Table 21 provides some examples of the training data as well as the resulting lemmas.

Training instance	Class	Resulting lemma
mandjietjies	Rtjies>	mandjie
vaderlandse	Re>	vaderlands
flens	0	flens

Table 21: Examples of lemmatisation training data

The feature selection, data representation and parameter optimisation that resulted in the best classifier as reported by Groenewald (2006) was used as the setting for all the classifiers trained in these experiments.

4.4.1.2 Task B: Developing an ASR system for Afrikaans

ASR systems are developed by using orthographically transcribed audio data as training data for systems such as the *Hidden Markov Model Toolkit* (*HTK*)[45], *CMU Sphinx*[46], *RWTH ASR*[47], etc. For purposes of this study, we used *HTK v3.4* (Young *et al.*, 2009).

The audio data which was used in this task was extracted from the same collection of news bulletins as described in Chapter 2 (see 2.4.2). All bulletins from female speakers were selected and consisted of 2,200 utterances (265 minutes of audio). This data was used as training and testing data for a speaker-independent, gender-specific system. We used Mel Frequency Cepstral Coefficients (MFCCs) as feature vectors with 12 MFCCs, energy and the first and second order derivatives of these (39 coefficients in total). Feature vectors were calculated for Hamming windowed speech frames with length 25ms extracted at 10ms intervals. This was used to train tied-state, context-dependent (tri-phone) HMMs consisting of three states with a standard “left-to-right” topology from a “flat start” initialisation. We performed seven mixture increments, resulting in a maximum of eight mixtures for the Gaussian Mixture Model (GMM) per state, using a diagonal covariance matrix. As the focus of this thesis is on resource-scarce languages, we assume that no additional resources are available for the task, and as such we used the transcriptions of the training and testing sets for the language model[48]. We used an Afrikaans pronunciation dictionary developed by Davel and de Wet (2010) as initial pronunciation dictionary, and used grapheme-to phoneme rules to generate missing pronunciations. The systems are classified by the recognition vocabulary as medium[49]. The systems contained 36 phonemes and a silence marker; see Annexure E for a list of these phonemes.

[45] http://htk.eng.cam.ac.uk

[46] http://cmusphinx.sourceforge.net/wiki

[47] http://www-i6.informatik.rwth-aachen.de/rwth-asr

[48] We used a bigram language model, generated with *HLStats*, by using transcriptions of the training and testing sets (of each increment). See Table 23 for more information on the increment sizes.

[49] Between 1,000 and 10,000 words (Whittaker & Woodland, 2001).

4.4.2 Data

In order to perform tenfold cross-validation, the data for both sets of experiments was organised as follows:

1. The gold standard data was randomised and divided into ten parts of equal size;
2. The ten parts were combined to produce ten sets of training data and ten sets of testing data; i.e. part one was used as test set one; parts two to ten were combined to produce training set one; part two was used as testing set two and parts one and three to ten were used as training set two, etc. This resulted in the training and testing sets for each of the ten folds;
3. From the training data of each of the folds, random subsets ranging from 10% up to 100% were extracted to produce ten increments per fold, thus each fold consisted of ten increments ranging from 10% of the training data up to 100% of the training data;
4. For each increment, errors were randomly generated according to results achieved by different groups of respondents in Chapter 2 (see 2.5.1 and 2.5.2) and Chapter 3 (see 3.5.1 and 3.5.2). This resulted in four training sets (Gold, Expert, Trained Novice and Untrained Novice) for each increment and for each fold, a total of 400 training sets.
5. To compare systems trained with the Gold data to double the quantity of data from the Expert, Trained Novice and Untrained Novice, a random subset (HGold) of 50% was extracted from each Gold training set. This was done to ensure that no ambiguity was present in the comparison with what would have been the case if, for example, the results of Gold increment one were compared to Expert increment two.

4.4.2.1 Error generation

In order to simulate real world levels of errors, the quality of annotations reported on in Chapter 2 (for expert and untrained novices[50]) and Chapter 3 (for trained novices) were used as the levels of errors that were generated in the different datasets.

[50] The results of the untrained novices and laymen in Chapter 2 had no statistical significant difference and as such, the results of the untrained novices were used.

4.4.2.1.1 Error generation for Task A

The expert respondent achieved an accuracy of 97% (see 2.5.1); the trained novice group achieved an average of 90% (see 3.5.1); and the untrained novice group achieved 75% (see 2.5.1).

Errors were generated as follows:

1. If a word had any class other than class 0, the class was randomly changed to either another random possible class (i.e. a class that could match the word based on the orthography), or to class 0.
2. For words that had a class 0, the class was changed to a random possible class.

Table 22 shows some examples of the generated errors and the resulting erroneous lemmas.

Training instance	**Correct class**	**Correct lemma**	**Generated class**	**Resulting lemma**
mandjietjies	Rtjies>	mandjie	Rs>	mandjietjie
vaderlandse	Re>	vaderlands	Rse>	vaderland
flens	0	flens	Rs>	flen

Table 22: Examples of generated errors

This resulted in four sets of data with varying levels of quality: data containing 0% errors (Gold), 3% errors (Expert), 10% errors (Trained Novice) and 25% errors (Untrained Novice). As mentioned above (see 4.4.2), a random subset (HGold) of 50% was extracted from each Gold training set. Table 23 provides an overview of the quantity of data in each fold, for each increment, and the number of generated errors in each set.

Increment	**Total words in Gold**	**Expert (3% errors)**	**Trained Novice (10% errors)**	**Untrained Novice (25% errors)**
1	6625	199	663	1655
2	13250	398	1326	3310
3	19875	597	1989	4965
4	26500	796	2652	6620
5	33125	995	3315	8275
6	39750	1194	3978	9930
7	46375	1393	4641	11585
8	53000	1592	5304	13240
9	59625	1791	5967	14895
10	66250	1990	6630	16550

Table 23: Number of words and errors in each increment

4.4.2.1.2 Error generation for Task B

As in Task A, three levels of errors were generated in the data according to results achieved by different groups of respondents in Chapter 2 and Chapter 3. Some of the errors made by respondents are not applicable to the actual training of the systems as all capital letters, noise markers and punctuation are removed before training the systems with *HTK*. Only the remaining categories of errors as discussed in Chapter 2 that can be present in the training data were randomly generated in the data. The category of compounding errors included both invalid compound composition (for example "*dieman*" (*theman*) instead of "*die man*" (*the man*)), and invalid compound decomposition (for example "*oor stuk*" (*ear piece*) instead of "*oorstuk*" (*earpiece*)). As compound decomposition decreases out-of-vocabulary words (and as a result increases system performance), only invalid compound compositions were generated in the compounding error category; only 50% of the total compounding errors were used. Insertions, deletions, substitutions and transpositions on the transcription level and spelling errors on the language level were also included. Table 24 provides an overview of the average annotated errors as reported in 2.4.5.2 (Expert and Untrained Novice) and 3.5.2 (Trained Novice) that were used to calculate the level of errors that were generated for each group. These errors were made on a total of 918 words.

	Expert	**Trained novices**	**Untrained novices**
Insertions	0	0.30	12.00
Deletions	0	2.90	41.70
Substitutions	1	4.40	19.30
Transpositions	0	0	0.20
Total transcription errors	1	7.3	73.2
Spelling errors	6	21.90	69.50
Compounding errors	3	7.70	18.80
Total language errors	9	29.6	88.3

Table 24: Annotated errors of expert, trained novices and untrained novices

Although the impact of transcription errors on ASR system performance will be different than the impact of language errors, both types of errors were generated in the same datasets as we investigate the overall impact on system performance of a combination of these two categories of errors. To illustrate the individual impact of these two categories of errors, we conducted an additional experiment where we randomly generated separate datasets for increment ten, and trained and evaluated these systems using tenfold cross-validation[51]. These results indicate that, in this specific experiment, and with these specific ratios of transcription and language errors, the language errors have a greater impact on system performance than transcription errors (see Annexure F). As the ratio of language errors is higher than the ratio of transcription errors, these results are only relevant to the systems trained in Experiment 1, and it cannot be extrapolated to any other ASR systems trained on different quantities or quality of data.

Using these levels of errors, we generated four sets of data: data containing 0% errors (Gold), 1.09% errors (Expert), 4.05% errors (Trained Novice) and 17.59% errors (Untrained Novice). Once again a random subset (HGold) of 50% was extracted from each Gold training set. Thus, fifty training sets for each fold were created, resulting in a total of 500 training sets consisting of 490,050 files and ten testing sets consisting of 2,200 files. Table 25 provides an overview of the data, with the number of utterances for each increment and the average number of words and generated errors in each set.

Increment	**Number of utterances per fold**	**Average Words per fold**	**Average Errors (Expert) (1.09% errors)**	**Average Errors (Trained Novice) (4.05% errors)**	**Average Errors (Untrained Novice) (17.59% errors)**
1	198	3743.80	40.81	151.62	658.53
2	396	7374.80	80.39	298.68	1297.23
3	594	11087.20	120.85	449.03	1950.24
4	792	14865.80	162.04	602.06	2614.89
5	990	18675.50	203.56	756.36	3285.02
6	1188	22281.20	242.87	902.39	3919.26
7	1386	26066.00	284.12	1055.67	4585.01
8	1584	29782.40	324.63	1206.19	5238.72
9	1782	33451.10	364.62	1354.77	5884.05
10	1980	37227.60	405.78	1507.72	6548.33

Table 25: Increment size and errors

[51] These results cannot be directly compared to the results in Experiment 1 or Experiment 2 as the errors were randomly generated in the datasets (e.g. a specific deletion was not generated in the same utterance and/or position in both datasets).

4.4.3 Evaluation criteria

The measures described in the following two sections were used to evaluate and compare all the systems trained in Experiment 1 and Experiment 2.

4.4.3.1 Evaluation criteria for Task A

Following Groenewald (2006), the generated training sets were used to train classifiers using *TiMBL* (Daelemans *et al.*, 2001). The classifiers of each increment for each fold were evaluated on the relevant testing set for the fold, i.e. all fifty classifiers of each fold were evaluated on the same testing set. *TiMBL* provides the accuracy of classifiers as a standard evaluation metric that is calculated by dividing the total correctly classified evaluation instances by the total evaluation instances. As each training instance consists of a word and the relevant class, the classifier accuracy is equal to the lemmatisation accuracy, and is the same as the accuracy metric used in Chapter 2 and Chapter 3 (i.e. we calculate the accuracy by dividing the total number of words correctly lemmatised and words correctly left unchanged, by the total words in the task). This measure was used to evaluate and compare all the systems trained in this experiment.

4.4.3.2 Evaluation criteria for Task B

We used *HTK v3.4* (Young *et al.*, 2009) to train all ASR systems on the generated data. The ASR systems of each fold were evaluated on the relevant testing set for the fold, i.e. all fifty classifiers of each fold were evaluated on the same testing set. *HTK* provides standard evaluation of recall of the systems. In addition to this measure, the WER was also calculated.

These measures were calculated according to the equations below:

$$Recall = \frac{n - s - d}{n}$$

$$WER = \frac{s + d + i}{n}$$

where

- *n* is the number of instances in the reference;
- *s* is the number of substitutions;
- *d* is the number of deletions; and
- *i* is the number of insertions.

4.4.4 Assumptions

For purposes of these experiments the associated costs, for example hourly rate of the annotators, the cost involved in training of annotators and the cost of setting up the project, protocols, equipment needed, etc. is assumed to be equal regardless of whether double-blind or single-base annotation is used. One aspect which could have an influence on cost is data collection, but the increase in cost of acquiring double the amount of data can be disregarded in cases involving text annotation as there is generally no major difference in collecting 10,000, 20,000 or 100,000 words. In the case of speech, however, the cost implication can be large. Depending on the domain, language and usage rights required by a project, it is possible to acquire audio data at a minimal cost or for free. Organisations such as the Dutch *TST-Centrale*, the *Resource Management Agency* (RMA) of South Africa and various other organisations also provide most of their resources free for research purposes. In this experiment we assume that there is no additional cost involved in acquiring more data.

For purposes of this experiment it is assumed that for the same effort (in terms of time and associated cost), single-base annotated data of varying quality can be annotated. It is also assumed that for the same amount of effort, only half this quantity can be annotated using double-blind annotation.

4.4.5 Hypotheses

The hypotheses for Experiment 1 were defined as follows:

- The null hypothesis was that all groups are equal, i.e. similar results can be obtained with data containing errors, and with data containing no errors.
- The alternative hypothesis was that all groups are not equal, i.e. system performance differs significantly.

For Experiment 2, the hypotheses were defined as follows:

- The null hypothesis was that all groups are equal, i.e. similar results can be obtained with more data containing errors, as with less data containing no errors.
- The alternative hypothesis was that all groups are not equal, i.e. system performance differs significantly.

4.5 Results, analysis and interpretation

Analysis of the results was performed using repeated measures ANOVAs to compare the average scores of different groups on a dependent variable. ANOVA relies on assumptions of normality of the data and homogeneity of variances. A Greenhouse-Geisser correction was done to correct for deviation from sphericity for the omnibus test for each increment and a *p*-value smaller than 0.05 was considered as sufficient evidence that the result was statistically significant. As in the previous two chapters, Cohen's *d*-value was used as a measure of practical significance. *d* = ±0.2 was considered a small effect (no practically significant difference), *d* = ±0.5 was considered a medium effect (practically visible difference) and *d* = ±0.8 was considered a large effect (practically significant difference).

4.5.1 Experiment 1

For Experiment 1, results from the systems trained with the Gold datasets were compared to results from systems trained with Expert, Trained Novice and Untrained Novice datasets in order to establish:

1. to what extent systems trained with the same quantity of data, but with varying levels of quality, is affected in terms of performance; and
2. if the quantity of the training data has an effect on the difference in system performance.

4.5.1.1 Task A: Lemmatisation of Afrikaans text data

4.5.1.1.1 Results

Table 26 and Figure 13 show the results of the systems trained with Gold, Expert, Trained Novice and Untrained Novice datasets. The results shown are average accuracy of the tenfold cross-validation per increment ranging from 10% of the training data up to 100%. As expected, the systems trained with Gold data achieved better results than the systems trained with the other data. The high scores of the systems trained with only 10% of the data can be attributed to the relative easy task of lemmatisation[52]. It is interesting to note that the differences in results were exceptionally small in the case of the systems trained with the Gold, Expert and Trained Novice data: an average difference over all ten increments of 0.08% worse for Expert data and 0.46% for Trained Novice data. The Expert data contained 3% generated errors and the data from Trained Novice contained 10% errors, but this scale of difference was not evident from the results of the systems trained on the data (as expected; see 4.4). The results of

[52] The low improvement in system performance of the Gold systems of only 5.79% between increment one and ten, even though ten times more data is used as training data, also shows that this task follows the 80/20 principle (also known as the Pareto principle).

the systems trained on Untrained Novice data show a larger decline in accuracy: an average of 3.83% worse than the systems trained on the Gold data, while the training data contained 25% errors. These results indicate that there is no linear relationship between the percentage of errors present in training data and the accuracy of the resulting systems.

Increment	Gold	Expert (3% errors)	Trained Novice (10% errors)	Untrained Novice (25% errors)
1 (10%)	86.637	86.472	85.549	80.848
2 (20%)	88.889	88.790	88.265	84.264
3 (30%)	90.022	89.948	89.621	85.963
4 (40%)	90.583	90.466	90.196	86.773
5 (50%)	91.159	91.053	90.793	87.507
6 (60%)	91.525	91.475	91.126	87.943
7 (70%)	91.827	91.774	91.496	88.650
8 (80%)	92.044	91.994	91.699	88.888
9 (90%)	92.306	92.249	91.963	89.011
10 (100%)	92.422	92.407	92.104	89.366

Table 26: Comparison of accuracy of systems trained with the same amount of data

When comparing the difference in performance of the Expert, Trained Novice and Untrained Novice systems as opposed to the Gold systems, one can also see that the differences decrease as the quantity of data increases. This decrease in difference of performance is most evident between the results of the Gold and Untrained Novice. In the first increment, the Untrained Novice system performed 5.79% worse than the Gold system, but in increment ten the difference is only 3.06% lower.

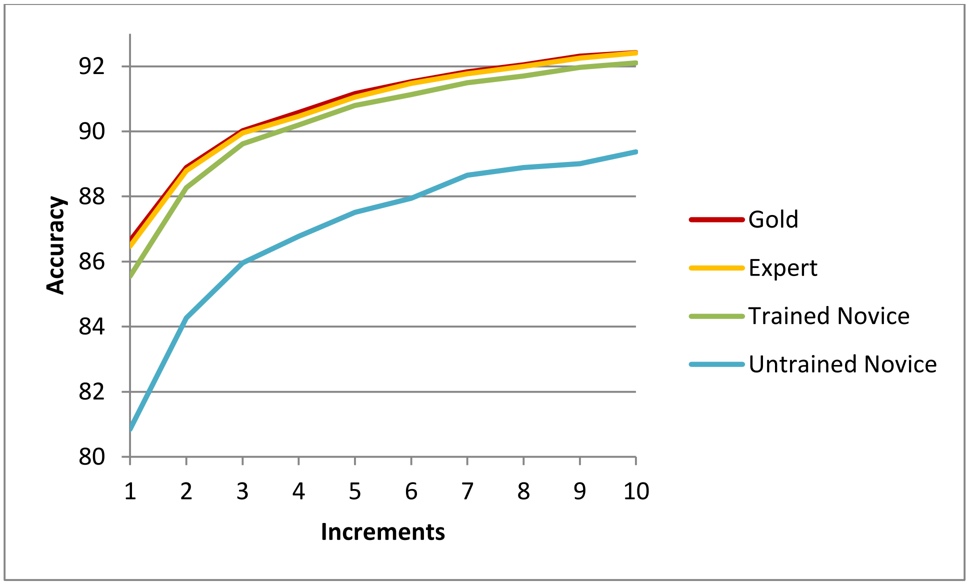


Figure 13: Accuracy of systems per increment

Figure 14 shows the average decrease in difference of system performance of all ten folds in terms of percentage when compared to the Gold system over all ten increments. The decrease in difference of performance when comparing the three systems to the Gold system regarding the increase in training data quantity, demonstrates that the effect of annotation errors in training data diminishes as the quantity of training data increases.

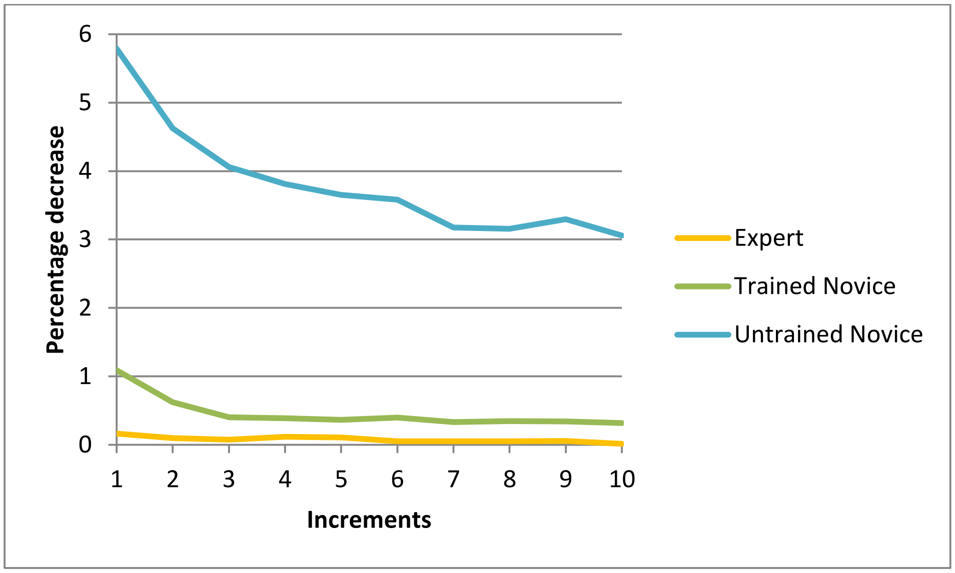


Figure 14: Difference in performance per increment

4.5.1.1.2 Statistical analysis and interpretation

Although the results in the previous section showed that errors in training data do not result in a linear decrease of system performance and that the decrease in performance lessens as the training data quantity increases, further investigations were conducted to determine the significance of these differences. Repeated measures ANOVAs were conducted to determine, for each increment, whether Gold, Expert, Trained Novice and Untrained Novice differed significantly or not. In Table 27 below, the means, standard deviations, 95% confidence intervals and Greenhouse-Geisser corrected *p*-values are reported for accuracy and for all increments.

Increment	Gold		Expert		Trained Novice		Untrained Novice		*P*-value
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Repeated measures ANOVA (Greenhouse-Geisser corrected)**
	(95% Confidence interval)		**(95% Confidence interval)**		**(95% Confidence interval)**		**(95% Confidence interval)**		
1	86.637	0.324	86.472	0.372	85.549	0.349	80.848	0.536	< 0.001
	(86.405; 86.869)		(86.206; 86.739)		(85.299; 85.798)		(80.464; 81.231)		
2	88.889	0.393	88.79	0.367	88.265	0.353	84.264	0.531	< 0.001
	(88.608; 89.170)		(88.527; 89.052)		(88.013; 88.518)		(83.884; 84.643)		
3	90.022	0.395	89.948	0.386	89.621	0.534	85.963	0.488	< 0.001
	(89.739; 90.305)		(89.672; 90.225)		(89.239; 90.003)		(85.614; 86.312)		
4	90.583	0.371	90.466	0.376	90.196	0.332	86.773	0.41	< 0.001
	(90.317; 90.848)		(90.197; 90.735)		(89.958; 90.433)		(86.479; 87.066)		
5	91.159	0.196	91.053	0.175	90.793	0.285	87.507	0.434	< 0.001
	(91.019; 91.299)		(90.928; 91.178)		(90.590; 90.997)		(87.197; 87.818)		
6	91.525	0.277	91.475	0.311	91.126	0.359	87.943	0.392	< 0.001
	(91.327; 91.723)		(91.253; 91.698)		(90.869; 91.383)		(87.663; 88.224)		
7	91.827	0.391	91.774	0.43	91.496	0.386	88.65	0.565	< 0.001
	(91.548; 92.106)		(91.467; 92.081)		(91.219; 91.772)		(88.246; 89.054)		
8	92.044	0.249	91.994	0.285	91.699	0.228	88.888	0.452	< 0.001
	(91.866; 92.222)		(91.790; 92.198)		(91.536; 91.862)		(88.564; 89.211)		
9	92.306	0.253	92.249	0.257	91.963	0.267	89.011	0.27	< 0.001
	(92.126; 92.487)		(92.066; 92.433)		(91.772; 92.154)		(88.818; 89.204)		
10	92.422	0.28	92.407	0.255	92.104	0.39	89.366	0.347	< 0.001
	(92.221; 92.622)		(92.225; 92.589)		(91.825; 92.383)		(89.117; 89.614)		

Table 27: Means, standard deviations and 95% confidence levels

Based on the ANOVA *p*-values (i.e. all $p < 0.001$), it is clear that there were differences in the performance of the various datasets for all iterations. Pairwise comparisons were conducted to explore where the differences lay. Table 28 shows the mean difference, standard error and *p*-values for each increment between the Gold, Expert, Trained Novice and Untrained Novice systems. Bonferroni corrections were performed on all *p*-values to compensate for multiple comparisons. For pairwise comparisons of all increments between Gold and Trained Novice, as well as between Gold and Untrained Novice, the *p*-values were < 0.05 (see Table 28), which indicates that for all increments of the Trained Novice and Untrained Novice systems, the performance achieved was statistically significantly lower than the performance of the Gold systems. This comparison also shows that the Gold systems did not perform better than the Expert systems on any increments.

Increment	Expert			Trained Novice			Untrained Novice		
	Mean difference	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**
1	0.164	0.808	0.499	1.088	< 0.001	3.406	5.789	< 0.001	13.779
2	0.099	1.000	0.274	0.623	< 0.001	1.761	4.625	< 0.001	10.437
3	0.073	1.000	0.200	0.401	0.001	0.900	4.059	< 0.001	9.638
4	0.117	0.978	0.330	0.387	< 0.001	1.159	3.810	< 0.001	10.272
5	0.106	1.000	0.601	0.365	0.001	1.577	3.651	< 0.001	11.432
6	0.050	1.000	0.179	0.399	< 0.001	1.312	3.582	< 0.001	11.125
7	0.053	1.000	0.136	0.331	0.016	0.898	3.177	< 0.001	6.893
8	0.050	1.000	0.197	0.345	< 0.001	1.523	3.157	< 0.001	9.117
9	0.057	1.000	0.236	0.344	0.002	1.390	3.295	< 0.001	13.275
10	0.015	1.000	0.059	0.318	< 0.001	0.987	3.056	< 0.001	10.217

Table 28: Pairwise comparison of Gold with Expert, Trained Novice and Untrained Novice systems

The *d*-values of the systems compared with the Gold systems (see Table 28) show similar results to the statistical significance. None of the increments of the Expert systems were practically significantly different from the Gold systems. All increments of the Trained Novice and Untrained Novice systems showed a practical significant difference.

The implication of these results is that, for the task of lemmatisation of Afrikaans, systems trained with the same quantity of data, but with a low level of errors (3% in the Expert data) are capable of achieving results that do not differ significantly (statistically or practically) from systems trained with the Gold data. The systems trained with Trained Novice and Untrained Novice data did, however, perform statistically and practically significantly lower. These results also indicate that it is more cost effective to use one expert to annotate data than to use two experts to double-blind annotate the same quantity of data, i.e. the annotation cost is halved, while the performance of the systems does not differ statistically or practically significantly.

4.5.1.2 Task B: ASR system for Afrikaans

As with Task A, results from the systems trained with the Gold data were compared with results from systems trained with the Expert, Trained Novice and Untrained Novice data. This comparison enabled us to establish to what extent performance of systems trained with the same quantity of data, but with varying levels of quality, was affected.

4.5.1.2.1 Results

Given the high WER[53] of the systems in increments one to four, only the results from increments five to ten are reported in this section. The WER and recall of all increments as well as the phone error rates (PER) of the systems are provided in Annexure G. Table 29 and Table 30 show the WER and recall of the different ASR systems trained with the Gold, Expert, Trained Novice and Untrained Novice data. As one would expect, the systems trained with the Gold data outperformed the other systems. The difference in performance was, however, less than the levels of errors present in the data.

Increment	**Gold** **(0% errors)**	**Expert** **(1.09% errors)**	**Trained Novice** **(4.05% errors)**	**Untrained Novice** **(17.59% errors)**
5 (50%)	70.11	70.67	71.37	76.65
6 (60%)	67.51	67.75	69.34	73.54
7 (70%)	65.56	66.19	67.24	71.99
8 (80%)	61.93	62.33	63.62	68.86
9 (90%)	60.32	60.89	62.05	67.26
10 (100%)	59.08	59.68	60.56	65.97

Table 29: WER of systems

Increment	**Gold** **(0% errors)**	**Expert** **(1.09% errors)**	**Trained Novice** **(4.05% errors)**	**Untrained Novice** **(17.59% errors)**
5 (50%)	69.40	69.07	68.69	65.49
6 (60%)	70.81	70.64	69.75	66.98
7 (70%)	72.05	71.59	70.87	67.84
8 (80%)	73.15	72.76	72.17	68.90
9 (90%)	73.86	73.56	72.80	69.91
10 (100%)	74.57	74.31	73.58	70.45

Table 30: Recall of systems

The WER scores, on average, of the systems trained with the Expert data (containing 1.09% generated errors) were only 0.50% worse and the Trained Novice systems (4.05% generated errors) were only 1.61% worse. The systems trained with the Untrained Novice data (17.59% generated errors) achieved 6.63% lower performance.

[53] The high WER is attributed to the small training sets as well as the small language models.

The difference in WER of the other three systems compared to Gold appeared to remain relatively constant, even though the quantity of data increased over the increments (see Figure 15). This lack of decrease in difference of performance illustrates the way in which the weight of annotation errors in ASR training data remains relatively unchanged even though the quantity of training data increases.

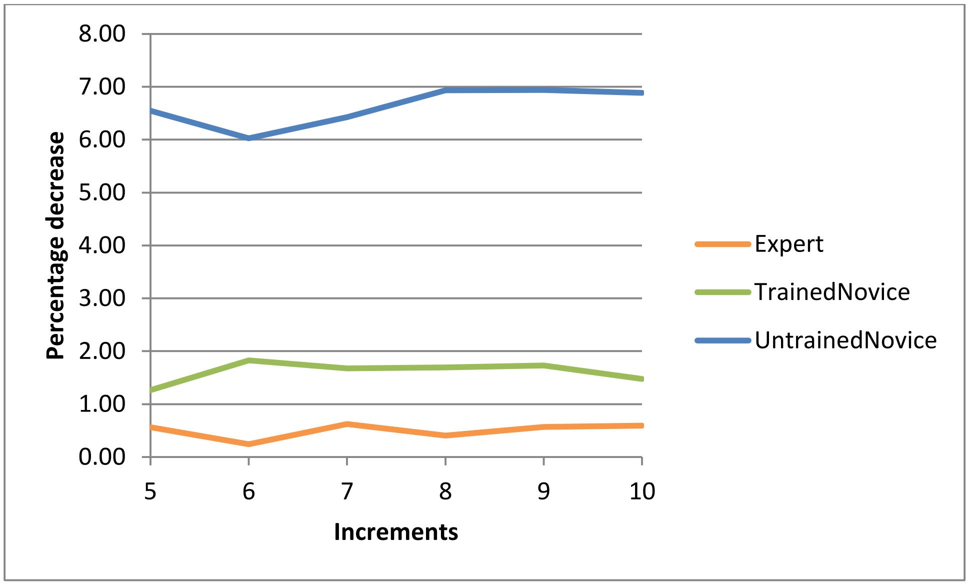


Figure 15: Difference in WER of systems compared to Gold systems

4.5.1.2.2 Statistical analysis and interpretation

Further investigations were conducted to determine the significance of the differences in performance of the systems. Repeated measures ANOVAs were conducted on the WER of the systems to determine, for each increment, whether Gold, Expert, Trained Novice and Untrained Novice systems differed significantly or not.

In Table 31 below, the means, standard deviations and 95% confidence intervals were reported for WER and for increments five to ten (see Annexure G for WER for all increments and recall). Once again, a Greenhouse-Geisser correction was done to correct for deviation from sphericity for the omnibus test for each increment.

Increment	Gold		Expert		Trained novice		Untrained novice		*P*-value
	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation	Repeated measures ANOVA (Greenhouse-Geisser corrected)
	95% Confidence interval		95% Confidence interval		95% Confidence interval		95% Confidence interval		
	Upper	Lower	Upper	Lower	Upper	Lower	Upper	Lower	
5	70.11	3.33	70.67	3.25	71.37	2.56	76.65	2.33	< 0.001
	72.17	68.05	72.69	68.66	72.96	69.79	78.10	75.21	
6	67.51	4.50	67.75	4.33	69.34	4.38	73.54	4.01	< 0.001
	70.30	64.72	70.43	65.07	72.05	66.62	76.02	71.05	
7	65.56	2.39	66.19	2.23	67.24	2.34	71.99	2.39	< 0.001
	67.04	64.08	67.57	64.80	68.69	65.79	73.47	70.51	
8	61.93	3.99	62.33	4.06	63.62	3.95	68.86	4.16	< 0.001
	64.40	59.45	64.85	59.82	66.07	61.17	71.44	66.29	
9	60.32	2.77	60.89	2.74	62.05	3.09	67.26	2.76	< 0.001
	62.04	58.61	62.59	59.20	63.97	60.14	68.97	65.56	
10	59.08	3.03	59.68	2.73	60.56	2.71	65.97	2.56	< 0.001
	60.96	57.21	61.37	57.99	62.24	58.88	67.56	64.38	

Table 31: Means, standard deviations and 95% confidence levels of WER

Based on the ANOVA *p*-values (i.e. all $p < 0.001$), it is clear that there were differences in the performance of the various datasets for these six increments. As before, pairwise comparisons were conducted to explore where the differences were. Table 32 shows the mean difference, *p*-values and *d*-values for increments five to ten of the systems of WER (see Annexure G for WER for all increments and for recall). Bonferroni corrections were performed on all *p*-values to compensate for multiple comparisons.

For pairwise comparisons of nine increments between Gold and Trained Novice and all increments between the Gold and Untrained Novice, the *p*-values were < 0.05 (see Table 32), which indicates that for all increments of the Trained Novice and Untrained Novice systems, the performance achieved was statistically significantly lower than the performance of the Gold systems. The pairwise comparison between the mean difference of the Gold and Expert systems showed no statistical significant differences.

Increment	**Expert**			**Trained novice**			**Untrained novice**		
	Mean difference	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**
5	0.564	1.000	-0.181	1.264	0.064	-0.449	6.546	< 0.001	-2.402
6	0.240	1.000	-0.057	1.826	< 0.001	-0.433	6.028	< 0.001	-1.491
7	0.626	0.785	-0.285	1.677	0.002	-0.748	6.428	< 0.001	-2.839
8	0.407	1.000	-0.107	1.693	0.002	-0.449	6.936	< 0.001	-1.794
9	0.569	1.000	-0.218	1.730	0.019	-0.622	6.938	< 0.001	-2.648
10	0.593	0.542	-0.217	1.474	0.001	-0.541	6.884	< 0.001	-2.589

Table 32: Mean difference, *p*-values and *d*-values of all increments of WER of systems compared to Gold

When we compare the *d*-values (see Table 32), the results indicate that none of the Expert systems were practically significantly different to the Gold systems. The Trained Novice systems differed practically significantly on one increment, and the Untrained Novice systems differed practically significantly on all increments. The ANOVA *p*-values, pairwise comparisons as well as the effect size of recall showed similar significance as the analysis of the WER (see Annexure G).

These results show that, for the task of ASR of Afrikaans, systems trained with the same quantity of data, but with a low level of errors (1.07% in the Expert data) were capable of achieving results that did not differ statistically or practically significantly from systems trained with the Gold data. The Trained Novice systems (4.05% errors) and the Untrained Novice systems (17.59% errors) showed statistical and practical significant differences. As with the results of the lemmatisation, the cost implication is that it is more cost effective to use one expert to annotate data than to use two experts to double-blind annotate the same quantity of data.

4.5.2 Experiment 2

In Experiment 2, systems were trained with double the quantity of Expert, Trained Novice and Untrained Novice data compared to the quantity of Gold data, to investigate whether the resulting systems would have a similar performance to Experiment 1, or whether the increase in quantity would benefit the performance of the systems to a larger extent than what the lower quality of the data decreases performance. As stated in 4.4.2, this experiment is performed to ensure that no ambiguity is present in the comparison as would have been the case if, for example, the results of Gold increment one are compared to the results of Expert increment two. Results from the systems trained with HGold (50% of Gold datasets, see 4.4.2) were compared to results from systems trained with Expert, Trained Novice and Untrained Novice datasets in order to establish:

- if a difference is present in performance of systems trained with double the quantity of data, but with varying levels of errors present in the data, when compared to systems trained with 50% of the quantity of the Gold data.

4.5.2.1 Task A: Lemmatisation of Afrikaans text data

4.5.2.1.1 Results

Table 33 and Figure 16 show the results of the systems trained with HGold, Expert, Trained Novice and Untrained Novice datasets. The results shown are average accuracy of the tenfold cross-validation, and per increment ranging from 10% of the training data up to 100%.

Increment	HGold	Expert	Trained Novice	Untrained Novice
1 (10%)	83.73	86.47	85.55	80.85
2 (20%)	86.75	88.79	88.27	84.26
3 (30%)	88.06	89.95	89.62	85.96
4 (40%)	88.87	90.47	90.20	86.77
5 (50%)	89.46	91.05	90.79	87.51
6 (60%)	89.95	91.48	91.13	87.94
7 (70%)	90.37	91.77	91.50	88.65
8 (80%)	90.62	91.99	91.70	88.89
9 (90%)	90.93	92.25	91.96	89.01
10 (100%)	91.09	92.41	92.10	89.37

Table 33: Accuracy of systems trained with the different amounts of data

The systems trained with HGold only achieved better results than the systems trained with the Untrained Novice data. Systems trained on the Expert and Trained Novice data achieved better results than systems trained with HGold data. Systems trained with the Expert data were on average 1.68% better and systems trained with the Trained Novice data were on average 1.30% better than the HGold systems. The increased performances were present on all ten increments of the experiment. The systems trained on the Untrained Novice data were on average only 2.06% worse than the systems trained on half of the Gold data, even though the data contained 25% errors.

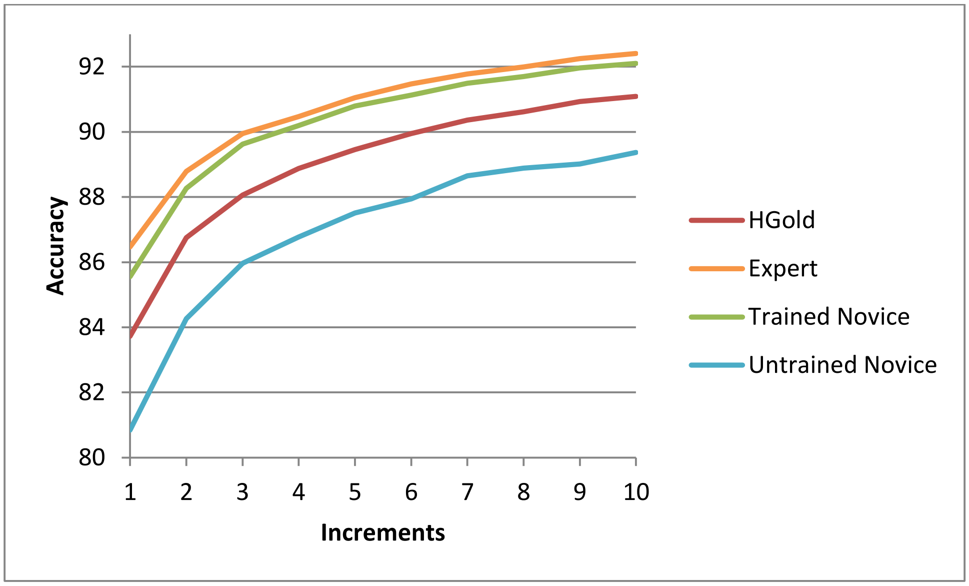


Figure 16: Accuracy of systems per increment

From increment one (containing 6625 instances) both the systems developed with Expert data (containing 3% errors) and Trained Novice data (containing 10% errors) outperformed the HGold system with 2.75% and 1.82% respectively. This increase in performance did lessen as the data quantity increased, and in increment ten the Expert systems achieved 1.32% higher performance and the Trained Novice systems 1.02% than HGold systems. The Untrained Novice systems (containing 25% errors) achieved 1.73% lower performance than the HGold systems.

These results show that although the quality of the data had a detrimental effect on the performance of the systems (viz. Experiment 1), the increase of quantity of data improved system performance to the extent that the Expert and Trained Novice systems outperformed the systems trained on 50% of the Gold data.

4.5.2.1.2 Statistical analysis and interpretation

In Experiment 1, the Expert systems showed no statistically significant difference in system performance when compared to the Gold systems, and the Trained Novice and Untrained Novice systems showed statistically significant lower performance. Although the results in 4.5.1.1.1 showed that the increase of quantity of data is more beneficial to the systems than what the quality of the data is detrimental to system performance, further statistical investigations were conducted to determine the significance of these differences.

In Table 34 below, the means, standard deviations and 95% confidence intervals are reported for accuracy and for all increments. As before, a Greenhouse-Geisser correction was done to correct for deviation from sphericity for the omnibus test for each increment.

Increment	**HGold**		**Expert**		**Trained Novice**		**Untrained Novice**		***P*-value**
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Repeated measure ANOVA (Greenhouse-Geisser corrected)**
	(95% Confidence interval)		**(95% Confidence interval)**		**(95% Confidence interval)**		**(95% Confidence interval)**		
1	83.726	0.417	86.472	0.372	85.549	0.349	80.848	0.536	< 0.001
	(83.428; 84.024)		(86.206; 86.739)		(85.299; 85.798)		(80.464; 81.231)		
2	86.754	0.351	88.79	0.367	88.265	0.353	84.264	0.531	< 0.001
	(86.503; 87.005)		(88.527; 89.052)		(88.013; 88.518)		(83.884; 84.643)		
3	88.064	0.325	89.948	0.386	89.621	0.534	85.963	0.488	< 0.001
	(87.832; 88.297)		(89.672; 90.225)		(89.239; 90.003)		(85.614; 86.312)		
4	88.874	0.444	90.466	0.376	90.196	0.332	86.773	0.41	< 0.001
	(88.556; 89.192)		(90.197; 90.735)		(89.958; 90.433)		(86.479; 87.066)		
5	89.455	0.35	91.053	0.175	90.793	0.285	87.507	0.434	< 0.001
	(89.205; 89.706)		(90.928; 91.178)		(90.590; 90.997)		(87.197; 87.818)		
6	89.954	0.373	91.475	0.311	91.126	0.359	87.943	0.392	< 0.001
	(89.687; 90.221)		(91.253; 91.698)		(90.869; 91.383)		(87.663; 88.224)		
7	90.365	0.349	91.774	0.43	91.496	0.386	88.65	0.565	< 0.001
	(90.116; 90.615)		(91.467; 92.081)		(91.219; 91.772)		(88.246; 89.054)		
8	90.621	0.363	91.994	0.285	91.699	0.228	88.888	0.452	< 0.001
	(90.361; 90.880)		(91.790; 92.198)		(91.536; 91.862)		(88.564; 89.211)		
9	90.932	0.43	92.249	0.257	91.963	0.267	89.011	0.27	< 0.001
	(90.625; 91.239)		(92.066; 92.433)		(91.772; 92.154)		(88.818; 89.204)		
10	91.089	0.34	92.407	0.255	92.104	0.39	89.366	0.347	< 0.001
	(90.846; 91.333)		(92.225; 92.589)		(91.825; 92.383)		(89.117; 89.614)		

Table 34: Means, standard deviations and 95% confidence levels

Based on the ANOVA *p*-values (i.e. all *p* < 0.001), it is clear that there were differences in the performance of the various datasets for all iterations. Based on pairwise comparisons of all increments between HGold and Expert, HGold and Trained Novice as well as between HGold and Untrained Novice, the *p*-values were < 0.05 (see Table 35), which indicates that for all increments of the Expert and Trained Novice systems, the performance achieved was statistically significantly higher than the performance of the HGold systems. For all increments of the Untrained Novice systems, the performance achieved was statistically significantly lower than the HGold systems.

Increment	**Expert**			**Trained Novice**			**Untrained Novice**		
	Mean difference	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**
1	-2.747	< 0.001	-7.325	-1.823	< 0.001	-4.998	2.878	< 0.001	6.318
2	-2.036	< 0.001	-5.977	-1.512	< 0.001	-4.525	2.490	< 0.001	5.831
3	-1.884	< 0.001	-5.566	-1.557	< 0.001	-3.713	2.101	< 0.001	5.342
4	-1.592	< 0.001	-4.079	-1.322	< 0.001	-3.555	2.101	< 0.001	5.182
5	-1.597	< 0.001	-6.088	-1.338	< 0.001	-4.419	1.948	< 0.001	5.208
6	-1.521	< 0.001	-4.669	-1.172	< 0.001	-3.375	2.010	< 0.001	5.540
7	-1.409	< 0.001	-3.793	-1.130	< 0.001	-3.240	1.716	< 0.001	3.850
8	-1.373	< 0.001	-4.435	-1.079	< 0.001	-3.749	1.733	< 0.001	4.456
9	-1.318	< 0.001	-3.919	-1.031	< 0.001	-3.036	1.921	< 0.001	5.640
10	-1.318	< 0.001	-4.623	-1.015	< 0.001	-2.924	1.724	< 0.001	5.287

Table 35: Pairwise comparison of HGold with Expert, Trained Novice and Untrained Novice systems

The *d*-values of the systems compared with the HGold systems (see Table 35) showed similar results to the statistical significance. All increments of the Expert, Trained Novice and Untrained Novice systems showed a practical significant difference.

These results imply that for the task of Afrikaans lemmatisation, for the same cost, it is more beneficial to use experts or trained novices to annotate more data than to use double-blind annotation. The results of the Untrained Novice systems showed that having a high level of errors in training data was detrimental to system performance, even if double the quantity of training data was used.

4.5.2.2 Task B: ASR system for Afrikaans

As with the task of lemmatisation, results from the systems trained with HGold were compared to results from systems trained with Expert, Trained Novice and Untrained Novice datasets. Given the small differences between the datasets in Experiment 1, it might be assumed that systems trained with double the quantity of data will perform better than systems trained with 50% of the Gold data, but the analyses are still performed to determine if this assumption holds, as well as to determine the significance of the differences.

4.5.2.2.1 Results

Table 36 and Table 37 show the WER and recall of the different ASR systems trained with the HGold, Expert, Trained Novice and Untrained Novice data. The results shown are averages of the tenfold cross-validation and per increment ranging from 50% of the training data up to 100%[54].

Increment	HGold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
5 (50%)	84.89	70.67	71.37	76.65
6 (60%)	79.95	67.75	69.34	73.54
7 (70%)	78.08	66.19	67.24	71.99
8 (80%)	74.84	62.33	63.62	68.86
9 (90%)	73.31	60.89	62.05	67.26
10 (100%)	70.93	59.68	60.56	65.97

Table 36: WER of systems

Increment	HGold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
5 (50%)	63.34	69.07	68.69	65.49
6 (60%)	65.23	70.64	69.75	66.98
7 (70%)	66.43	71.59	70.87	67.84
8 (80%)	67.52	72.76	72.17	68.90
9 (90%)	68.57	73.56	72.80	69.91
10 (100%)	69.69	74.31	73.58	70.45

Table 37: Recall of systems

When comparing the average WER of the systems trained with the HGold data to the Expert, Trained Novice and Untrained Novice systems, all of these systems performed better (see Figure 17). The systems trained on the Expert data performed on average 12.41% better, Trained Novice 11.30% better

[54] As with Experiment 1, only the results from increments five to ten are reported in this section. The WER and recall of all increments, as well as the PER of the systems are provided in Annexure H.

and the Untrained Novice systems 6.29% better. The increased performances were present on all ten increments of the experiment. Even the systems trained on the Untrained Novice data that contained 17.59% generated errors showed an average of 6.29% higher performance than the systems trained on half of the Gold data.

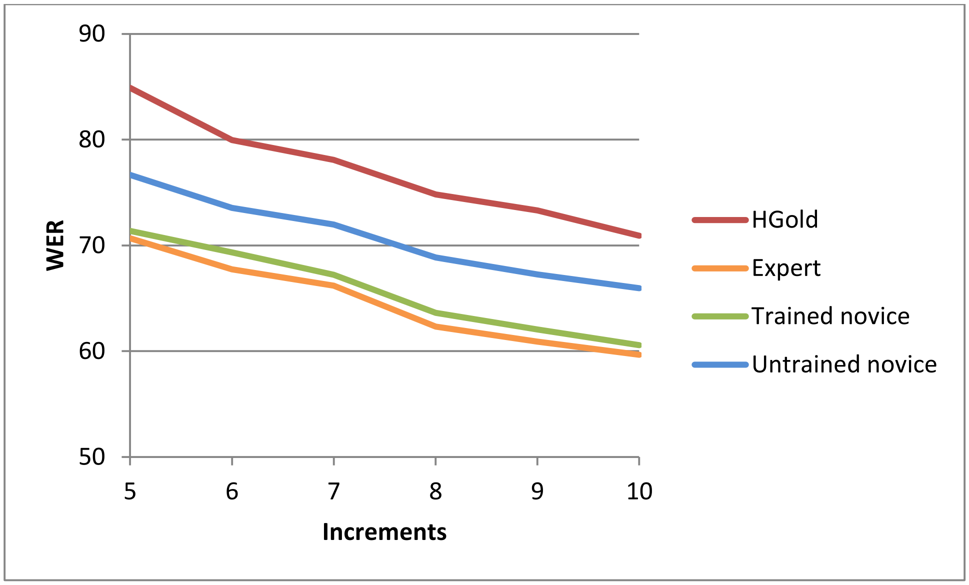


Figure 17: Average WER of systems

4.5.2.2.2 Statistical analysis and interpretation

As before, further statistical investigations were conducted to determine the significance of these differences. Repeated measures ANOVAs were conducted to determine, for each increment, whether HGold, Expert, Trained Novice and Untrained Novice differed significantly or not. In Table 38 below, the means, standard deviations and 95% confidence intervals are reported for WER for increments five to ten (see Annexure H for WER for all increments and recall). As usual, a Greenhouse-Geisser correction was done to correct for deviation from sphericity for the omnibus test for each increment.

Based on the ANOVA *p*-values, it is clear that there were differences in the performance of the various datasets for all iterations. As in Experiment 1, pairwise comparisons were conducted to explore where the differences were. For pairwise comparisons of WER of increments five to ten (see Annexure H for all increments and recall) between HGold and Expert systems as well as between HGold and Trained Novice systems, and between HGold and Untrained Novice systems, the *p*-values were < 0.05 (see Table 39), which indicates that for all six increments of the Expert, Trained Novice and Untrained Novice systems, the performance achieved was statistically significantly higher than the performance of the HGold systems.

Increment	**HGold**		**Expert**		**Trained novice**		**Untrained novice**		***P*-value**
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Repeated measures ANOVA (Greenhouse-Geisser corrected)**
	95% Confidence interval		**95% Confidence interval**		**95% Confidence interval**		**95% Confidence interval**		
	Upper	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	
5	84.887	3.375	70.673	3.249	71.373	2.557	76.655	2.33	< 0.001
	86.979	82.795	72.686	68.659	72.957	69.788	78.099	75.21	
6	79.955	4.142	67.75	4.329	69.336	4.382	73.538	4.011	< 0.001
	82.522	77.387	70.433	65.067	72.052	66.62	76.023	71.052	
7	78.081	4.255	66.187	2.234	67.238	2.336	71.989	2.386	< 0.001
	80.718	75.443	67.572	64.802	68.686	65.79	73.468	70.51	
8	74.836	3.907	62.335	4.057	63.621	3.951	68.864	4.16	< 0.001
	77.257	72.414	64.849	59.82	66.07	61.172	71.442	66.286	
9	73.311	2.96	60.894	2.736	62.055	3.087	67.263	2.755	< 0.001
	75.146	71.477	62.59	59.198	63.968	60.141	68.971	65.555	
10	70.929	3.725	59.677	2.725	60.558	2.711	65.968	2.561	< 0.001
	73.238	68.621	61.367	57.988	62.238	58.878	67.556	64.381	

Table 38: Means, standard deviations and 95% confidence levels for WER

The *d*-values (see Table 39), indicate that all of the Expert, Trained Novice and Untrained Novice systems were practically significantly different to the Gold systems. The ANOVA *p*-values, pairwise comparisons as well as the effect size of recall showed similar significance as the analysis of the WER (see Annexure H).

	Expert			**Trained novice**			**Untrained novice**		
Increment	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**
5	14.217	< 0.001	4.523	13.517	< 0.001	4.758	8.237	< 0.001	2.992
6	12.205	< 0.001	3.037	10.615	< 0.001	2.625	6.415	< 0.001	1.659
7	11.891	< 0.001	3.689	10.841	< 0.001	3.330	6.091	< 0.001	1.862
8	12.506	< 0.001	3.309	11.216	< 0.001	3.009	5.976	< 0.001	1.560
9	12.421	< 0.001	4.592	11.261	< 0.001	3.923	6.051	< 0.001	2.230
10	11.249	< 0.001	3.634	10.369	< 0.001	3.356	4.959	< 0.001	1.636

Table 39: Mean difference and *p*-value of all increments of WER on word level of systems compared to HGold

These results indicate that for the same cost, it is more beneficial for the task of Afrikaans ASR to use experts, trained novices or untrained novices to annotate more data than to use double-blind annotation. The results of the Untrained Novice systems also showed that even a high level of errors in training data was not detrimental to system performance if double the quantity of training data was used.

4.6 Conclusion

The aim of this chapter was to establish whether it is more beneficial to focus on the quality or the quantity of training data. In order to establish this, two sets of experiments were performed, for two distinct tasks.

In Experiment 1 we compared systems trained on the same quantity of data, but with varying quality, to determine the effect of errors in annotated data on systems trained with the data. For lemmatisation, four sets of data containing 0% errors (Gold), 3% errors (Expert), 10% errors (Trained Novice) and 25% errors (Untrained Novice) were generated. For ASR, four sets of data containing 0% errors (Gold), 1.09% errors (Expert), 4.05% errors (Trained Novice) and 17.59% errors (Untrained Novice) were generated. These sets of data were divided into ten increments, ranging from 10% to 100% of the data. Systems trained with these datasets were compared to establish the practical implication on systems trained with erroneous data.

Results for both tasks showed that there were no statistically or practically significant differences in performance of systems trained on the Gold data when compared to systems trained with the Expert data. Results of the systems trained with the Trained Novice as well as the Untrained Novice data did, however, show a statistical and practical significant difference.

Two observations are evident from the results. Firstly, performance of systems trained with low levels of errors, i.e. the Expert data, was comparable to performance of systems trained on Gold data. For the task of lemmatisation, the Expert systems showed a decrease of only 0.08% on performance, while the ASR systems trained on the Expert data showed a decrease of 0.36% on system performance.

Secondly, the decrease of system performance was not linear to the levels of errors present in the data. For lemmatisation, the Expert systems achieved 0.08% lower system performance, the Trained Novice systems 0.46% lower and the Untrained Novice systems 3.38% lower – but these systems were trained on data containing 3%, 10% and 25% errors respectively. Similar results were shown for the ASR systems: the Expert data (containing 1.09% generated errors) were only 0.57% lower and the Trained Novice systems (4.05% generated errors) were only 1.62% lower. The systems trained with the Untrained Novice data (17.59% generated errors) achieved 6.38% lower performance.

In Experiment 2, a subset of 50% was extracted from the Gold datasets. Results from systems trained on these HGold datasets were compared to results from the relevant increments of the Expert, Trained Novice and Untrained Novice systems. This enabled us to compare systems trained with double the quantity of data but with varying levels of errors, to systems trained with Gold data. This was done to establish whether the benefits of increasing the quantity of data would outweigh the detrimental effect of errors on system performance. By performing this comparison, we were able to make a recommendation concerning the decision of whether to focus on increasing the quality of annotated data (i.e. by implementing quality control, specifically using double-blind annotation), or to focus on increasing the quantity of annotated data (i.e. by annotating additional data).

Results from the lemmatisation systems showed that systems trained on the Expert data (containing 3% errors) as well as the Trained Novice data (containing 10% errors) outperformed systems trained on the HGold data. The increase of performance in both cases was statistically and practically significantly better. The systems trained on the Untrained Novice data (containing 25% errors) did, however, perform statistically and practically significantly lower than the HGold systems. Results of the ASR systems showed that the Expert (containing 1.09% errors), Trained Novice (containing 4.05% errors), as well as the Untrained Novice (containing 17.59% errors) systems outperformed the HGold systems. The increase in performance was statistically and practically significant for all of these systems.

The results from Experiment 1 showed that errors in training data are detrimental to system performance, but the decrease in performance is not linear to the levels of errors present in the data. Results from Experiment 2 showed that it is more beneficial to use experts or trained novices to annotate more data than to use double-blind annotation. The results of the Untrained Novice ASR systems showed that even a high level of errors in ASR training data is not detrimental to system performance if double the quantity of training data is used. The lemmatisation systems trained on the Untrained Novice data were the only systems that did not outperform the systems trained on the HGold data, but the Untrained Novice data contained 25% errors.

The cost implications of these results are as follows:

- For half the cost, one expert can annotate the same quantity of data as would be annotated by using double-blind annotation and the systems do not differ significantly in terms of performance (for lemmatisation and ASR of Afrikaans); and

- For the same cost, double the quantity of data can be annotated by an expert or trained novice (for lemmatisation and ASR) and by an untrained novice (for ASR), as that which could be annotated using double-blind annotation and the performance of resulting systems are significantly higher.

From the results and analysis of both tasks in both experiments as well as from the cost implication, it is evident that it is more beneficial to focus on the quantity of training data than on its quality. This confirms the maxim that more data is better data.

5 Chapter 5: Conclusion

5.1 Summary

The development of linguistic data is imperative for the HLT enablement of any language. Given the limitations of available resources, finances and expertise of resource-scarce languages, projects entailing HLT development for resource-scarce languages explore and implement various strategies through which the development of HLT resources can be expedited.

Although strategies are implemented in various projects, the efficiency of some of these strategies in creating resources for resource-scarce languages is not always clearly established. Thus, the main aim of this research was to determine which strategies are the most efficient for developing resources for HLTs for resource-scarce languages. Secondary to the main aim was to make recommendations regarding which of these methods should be implemented in practice.

The first chapter provided a brief overview of some of the strategies implemented in HLT development for mainstream languages as well as problems related to the implementation of these strategies for resource-scarce languages. The scope of this study was limited to factors related to the annotator and his/her environment, and specific aims were formulated:

1. To compare the results obtained using experts and non-experts for the task of linguistic annotation of data for resource-scarce languages in order to establish whether non-experts are a suitable alternative for annotation;
2. If comparable results can be obtained using non-experts, to establish whether it is beneficial to use novice annotators instead of laymen;
3. To establish the benefits in terms of time and quality when using tailor-made software instead of domain-specific or general-purpose software;
4. To establish whether additional development time of tailor-made software can be justified by the savings in annotation time; and
5. To establish if it is more beneficial to focus on the quality or the quantity of training data.

Specific research questions for each factor (i.e. relating to annotators, user interfaces, and quality vs. quantity) were also posed.

In Chapter 2, it was shown that no suitable workforce for Afrikaans (as well as other resource-scarce languages) was available via traditional crowdsourcing channels. Given the limited number of linguistic experts available, it was still prudent to investigate if a crowd-like group (i.e. untrained, recruited respondents), could be used for annotation of data for resource-scarce languages. To investigate the suitability of non-experts for annotation, untrained non-experts were sourced to complete the tasks. We sourced three groups of respondents, who were all native speakers of Afrikaans, with different levels of expertise: two experts, twenty novices (undergraduate students studying for a bachelor's degree with the subject Afrikaans included in his/her curriculum) and twenty laymen (people who had never studied Afrikaans at tertiary level) to determine the influence on performance of these different skill levels. We conducted systematic experiments where the respondents were tasked with providing lemmas for 1,000 words and providing orthographic transcriptions of six minutes of audio data.

For the task of lemmatisation, no significant differences were visible in the comparison of average time taken to complete the task. The expert achieved an accuracy of 97.10%, while the novice and laymen groups achieved 75.64% and 78.08% respectively. The accuracies of both groups were statistically significantly lower than the accuracy achieved by the expert. For the task of orthographic transcription of audio data, the comparison of average time taken to complete the task also showed no significant differences. A breakdown of the errors into transcription, language and protocol errors showed statistically significant differences between the expert and both the novices and laymen. The expert made a total of 29 errors, while the novices made an average total of 267.4 errors and the laymen made an average of 162.8 errors.

From the results it is evident that the experts outperformed the non-experts on both tasks, and that the differences in performance are significant. No significant difference between the novices and laymen was evident. We can therefore answer research questions 1 and 2 as follows:

1. For the task of linguistic annotation of data for resource-scarce languages, results similar to those achieved by experts cannot be obtained by using non-experts.
2. There is no measurable benefit in using novice annotators instead of laymen.

In Chapter 3, we conducted systematic experiments where the respondents completed the same tasks as in the previous chapter, but in different software environments. For the task of lemmatisation, forty respondents completed the task in four software environments and the results showed that the respondents who used the tailor-made software completed the task statistically significantly faster than respondents using the general-purpose software (a saving of 49.66% vs. *CrowdFlower* and 52.67% vs. *Excel*) and the domain-specific software (a saving of 46.54% vs. *LARALite*). The accuracy achieved by respondents in *LARAFull* (93.92%) was better than the accuracy achieved in *CrowdFlower* (90.33%), *Excel* (88.63%) and *LARALite* (90.92%).

For the task of orthographic transcription of audio data, thirty respondents completed the task in three software environments and the results showed that the respondents who used the tailor-made software also completed the task in a statistically significantly faster time than the respondents using the other software environments (a saving of 20.65% compared to *CrowdFlower* and 14.25% when compared to *Praat*). The data annotated in the tailor-made software also contained statistically significantly fewer errors than the data annotated in the other software environments, especially with regard to spelling errors (*CrowdFlower* = 219; *Praat* = 234; *TARA* = 126) and protocol errors (*CrowdFlower* = 162; *Praat* = 171; *TARA* = 23).

Based on these results, we can answer research question 3 as follows:

3. It is beneficial in terms of time and quality to use tailor-made software instead of domain-specific or general-purpose software.

In order to establish whether the additional development time of tailor-made software could be justified by the savings in annotation time, we extrapolated the results of Tasks A and B to real world annotation projects, which included the annotation of two million tokens and the transcription of 550 hours of audio data. Although the comparison showed that, in the case of these two specific projects, the benefit of the saving of annotation time justified the development time, we concluded that the scope of a project has to be in excess of one million tokens or 300 hours of audio data for development time to be justified. Because of the small scope of projects entailing the development of resources for resource-scarce languages, we can answer research question 4 as follows:

4. In the context of linguistic annotation of data for resource-scarce languages, the additional development time of developing tailor-made software cannot be justified by the savings in annotation time.

In Chapter 4, two sets of experiments for the tasks of developing a lemmatiser and ASR system for Afrikaans were performed in order to establish whether it is more beneficial to focus on the quality or the quantity of training data. In Experiment 1 we compared systems trained on the same quantity of data but with varying quality, to determine the effect of errors in annotated data on systems trained with the data. Errors were generated on three levels on different datasets, and systems were trained with these datasets to establish the practical implications on systems trained with erroneous data.

Results for Task A showed that there were no statistically significant differences in accuracy of systems trained on the Gold data when compared to systems trained with the Expert data (with an average of 0.08% worse for Expert data over all ten increments). Results of the systems trained with the Trained Novice data (0.46% worse) and Untrained Novice data (3.83% worse) did, however, show a statistically significant difference. Results for Task B showed that there were no statistically significant differences in WER of systems trained on the Gold data when compared to systems trained with the Expert data (with an average over all ten increments of 0.50% worse), and Trained Novice data (1.61% worse). Results of the systems trained with the Untrained Novice data (6.63% worse) showed a statistically significant difference on WER.

In Experiment 2, a subset of 50% (HGold) was extracted from the Gold datasets to enable us to compare systems trained with double the quantity of data but with varying levels of errors, to systems trained with HGold data. Results from the lemmatisation systems showed that systems trained on the Expert (on average 1.68% better) as well as the Trained Novice data (1.30% better) outperformed systems trained on the HGold data. The increase in performance in both cases was statistically significantly better. The systems trained on the Untrained Novice data (2.06% worse) did, however, perform statistically significantly lower than the HGold systems. Results of the ASR systems showed that the Expert (average of 12.41% better), Trained Novice (11.30% better), as well as the Untrained Novice systems (6.29% better) outperformed the HGold systems. The increase in performance was statistically significant for all of these systems. Based on the comparisons in both experiments, we can answer research question 5 as follows:

5. For these particular tasks, it is more beneficial to focus on the quantity than on the quality of training data.

Based on these answers, we provide some recommendations for future projects related to resource creation for resource-scarce languages in the following section.

5.2 Recommendations

In this section we translate the answers from the previous section into recommendations to future projects related to resource-scarce languages. Based on our past experience with resource-scarce languages, we know that it is often the case that the availability of potential annotators is limited, and such annotators have mostly word processing skills in a graphical user interface (GUI) environment. In worst cases, annotators sometimes have difficulties with file management, unzipping, proper encoding of text files, and so forth. This has an impact on aspects such as recruitment of annotators, training, GUI design, etc. These aspects are not taken into account in the recommendations in this section as we assume that such aspects have been addressed in advance (e.g. by large scale recruitment efforts, training in computer literacy and designing user-friendly environments) by the project. Although the scope of this study was limited to Afrikaans and to two intermediate linguistic tasks, we are of the opinion that these results are not only applicable to Afrikaans and the two specific tasks, but also to other resource-scarce languages and other tasks of similar complexity.

Let us create two hypothetical scenarios. In scenario 1, a project aims to develop high quality annotated data (gold standard) for a resource-scarce language, for use in future research-orientated projects. Scenario 2 entails the development of core technologies for a resource-scarce language.

The first choice for these projects is what the skill level of the annotator should be. According to the results obtained in Chapter 2, experts outperformed novices and laymen in terms of the quality of the annotated data, and no significant differences were evident between the quality of data annotated by novices or laymen. For scenario 1, the project should use expert annotators to ensure high quality annotations. For scenario 2, results from Chapter 4 indicate that systems trained on expert data are not statistically significantly different to systems trained on gold standard data, but the systems are statistically significantly different to systems trained on untrained novice data. Chapter 3, however, showed that trained novices (10% errors for lemmatisation; 4.05% for orthographic transcriptions) using the same software environment as the untrained novices (25% errors for lemmatisation; 17.57% for orthographic transcriptions) achieved results that were more similar to results achieved by the experts (3% errors for lemmatisation; 1.09% for orthographic transcriptions). Chapter 4 also showed that double the quantity of trained novice data compared to gold standard data resulted in systems that achieved similar and even better performance. If we consider that the professional fees of experts are usually considerably higher than the fees of novices, and as a result we can annotate more data using novices than experts, the project in scenario 2 could use trained novices to annotate the data.

Next, these projects should decide on the software to use. Chapter 3 showed that it is beneficial to use tailor-made software instead of general-purpose or domain specific software, both in terms of time needed to complete the tasks, and quality of the annotated data. Thus, if tailor-made software is available, the projects should use it. However, if tailor-made software has to be developed, the development time (and associated cost) can only be justified if the scope of the project is large enough. Chapter 3 showed that one million tokens need to be annotated – or 300 hours of audio data need to be transcribed – to justify the development time. The project in scenario 1 should also take the improved quality into account, as the goal is to annotate data of as high a quality as possible. In Chapter 3, trained novice respondents using tailor-made software achieved 3.59% higher accuracy in the task of lemmatisation than the trained novices using general-purpose software. If the development time needs to be justified, i.e. if only a limited budget is available, then the projects in both scenarios should not develop tailor-made software. One important factor to consider is that existing software can be customised in a much shorter time. For purposes of this study, the savings in annotation time was compared to the development time of *LARA2*, in order to determine whether the development of tailor-made software from scratch can be justified by the savings in annotation time. If we consider that the customisation of *LARAFull* from *LARA2* only took thirty hours of development time, the savings in annotation time of 75,000 tokens can justify the development time.

The last choice which the projects need to make is whether to focus on data quality or on data quantity. For the project in scenario 1, the clear choice is to focus on the quality. The project should use double-blind annotation with adjudication, or some method of automatically identifying possible errors (not discussed in this study). Results from Chapter 4 showed that the project in scenario 2 should rather focus on the quantity of the data, by single-annotating more data. Systems trained on double the quantity of data containing errors, outperformed systems trained on half the quantity of gold standard data.

In summary: for scenario 1 – a project that aims to develop gold standard data – the project can obtain the highest quality annotations by using experts to double-blind annotate data, in tailor-made software (if provided for in the budget or if the development time can be justified by the savings in annotation time). For scenario 2 – a project that aims to develop a core technology – the project should use experts or trained novices to single-annotate data in tailor-made software (if provided for in the budget or if the development time can be justified by the savings in annotation time).

5.3 Future work

Based on the results of this study, we have identified several areas which could be further investigated. In Chapter 2, the results showed that laymen outperformed novices, although the differences were not statistically or practically significant. This contradicts what one would expect, as the novices were undergraduate students with the language of Afrikaans included in their curriculums. From the error analyses, we could speculate as to the reason, but this anomaly should be further investigated to enable us to definitively determine why the laymen outperformed the novices.

From the results of untrained novices in Chapter 2 and the results of the trained novices in Chapter 3, it is evident that training has an immense influence on the performance of the annotators. The respondents in Chapter 3 only received one hour of training, including training in the software they used as well as the task. For the task of lemmatisation, the quality improved by 15%, while for the task of orthographic transcription of audio data, the quality improved by 13%. Further work should be done with more intensive training, for example using various iterations of training and feedback, to investigate how training can benefit performance of annotators. It should also be investigated whether more intensive training will result in quality of data annotations that are similar to the quality of annotations achieved by experts.

With regard to tailor-made software, an area that requires further investigation is the inclusion of methods aimed at expediting annotation time and quality, for example bootstrapping. By implementing such methods to provide the annotator with pre-populated data, the time needed for annotation can be greatly reduced and the quality can be improved. Bootstrapping can be integrated into the process or even in the software by retraining a system on data as the annotator progresses, and thereby improving the quality of the pre-populated data provided to the annotator (Davel & Barnard, 2005; Van Huyssteen & Puttkammer, 2007). It would be interesting to determine which quality level of pre-populated data would be beneficial to the annotator, since low levels of quality could actually have a detrimental effect.

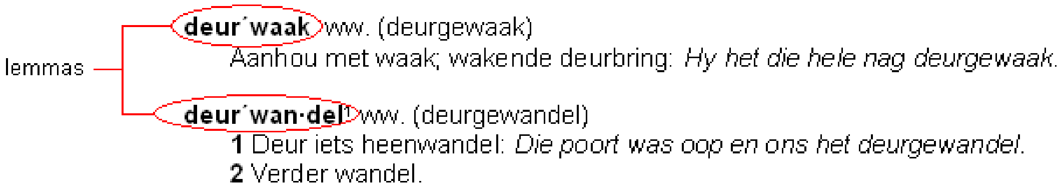
Following the annotation process, some aspects of the training of core technologies may be further investigated, such as the effect of different errors on systems. For the task of lemmatisation, errors were made by respondents not lemmatising words which needed to be lemmatised, derivations being lemmatised instead of leaving the words as they originally appeared, capitalisation errors, spelling errors and empty responses. Data could be generated to only include one of these types of errors, to determine its effect on the systems. Similar experiments could be conducted with ASR systems by generating only transcription, language or protocol errors. By determining the effect of different types of errors, measures could be implemented in future to reduce the types of errors with the most detrimental effect on systems. By identifying error types that do not have a major impact on systems (for example punctuation in ASR data is removed before training a system) the protocol of a task can also be simplified. This might be beneficial to the quality of annotations, as annotators can focus on relevant stipulations.

Finally, these experiments could be duplicated with other languages as well as with other tasks to establish whether the same conclusions and recommendations would be applicable. The experiments in this study can easily be duplicated by using the setups and methodologies described, especially the comparison of systems as described in Chapter 4.

Annexure A

Protocol: Lemmatisation of Afrikaans text data

For the purposes of this task, a lemma is defined as *the simplest form of a word as it would appear as headword in a dictionary*[55]*.* The aim of lemmatisation is to identify the lemma of inflected words, and for the purposes of this study lemmatisation is defined as t*he process of normalising all inflected forms of a lexical word to its common lemma.*



deur´waak ww. (deurgewaak)
Aanhou met waak; wakende deurbring: *Hy het die hele nag deurgewaak.*

deur´wan·del¹ ww. (deurgewandel)
1 Deur iets heenwandel: *Die poort was oop en ons het deurgewandel.*
2 Verder wandel.

Figure 18: Example from *Handwoordeboek van die Afrikaanse Taal* (HAT) (Odendal & Gouws, 2005)

For Afrikaans, we identify only the following categories of inflection:

Nouns

1. Plural (e.g. *tafel**s***)
2. Diminutive (e.g. *tafel**tjie***)

Adjectives

3. Comparative degree (e.g. *mooi**er***)
4. Superlative degree (e.g. *mooi**ste***)
5. Attributive *-e* (e.g. *interessant**e** persoon*)
6. Partitive genitive (e.g. *iets mooi**s***)

Verbs

7. Infinitive *-e* (e.g. *iets te drink**e***)
8. Past tense (e.g. ***ge**skop*)
9. Present participle (e.g. *skreeu**end***)

[55] In morphological terms, it is referred to as the base form, base or canonical form.

10. Weak past participle *ge-...-t/-d* (e.g. ***ge****meganiseer****d***)

The following categories of verb conjugations could also be identified, but are not used for lemmatisation in this task (in other words, words in these categories remain as they are). They usually consist of the prefixes ***ge-, be-, her-, er-, ont-*** and ***ver-***.

11. Strong past participle (finite list, e.g. ***geswore***)
12. Inchoative (e.g. ***ont****vlam*)
13. Intensive (e.g. ***ver****slaap*)
14. Repetitive (e.g. ***her****doop*)
15. Transitive (e.g. ***be****vaar*)

In addition, some extra stipulations are made in the protocol used in this task:

- You are provided with tokenised sentences (i.e. one word per line) and must provide the lemma for each word.
- If a word is already a lemma, provide the word exactly as it originally appeared.
- If only punctuation occurs, nothing should be entered into the textbox; it should be left blank.
- Words that appeared at the beginning of sentences should be provided with an initial lower-case letter, except if the word is a proper name.

Annexure B

Protocol: Orthographic transcription of Afrikaans audio data

For the purposes of this task, orthographic transcription is defined as *writing down verbatim what is heard in an audio recording, irrespective of incorrect sentence structures or grammatical errors present in the speech. No changes whatsoever are made to what a speaker says.*

In addition, some extra stipulations are made in the protocol used in this task:

Spelling

- Words have to be spelled according to the conventions used in a dictionary for the language.
- If you are uncertain about the spelling of a name, add a question mark in brackets *(?)* directly after the name followed by a space (e.g. *Gadhafi(?) said that...*)[56]*.*

Capital letters

- Contrary to spelling conventions, words at the beginning of a sentence should be written in lower case, except if the first word of a sentence is a name.
- Following spelling conventions, proper nouns, titles of books, place-names, brand names, names of societies, commissions, etc. should to be written with an initial upper-case letter.
- Multi-word named entities should to be written with an initial upper-case letter for each word (e.g. *North American Space Association*).

Numbers, letters and abbreviations

- Write out ordinal numbers and numbers that make out part of a word instead of using digits (i.e. "*twelve*" and not "*12*").
- If an abbreviations is heard in the audio recording, the letters of the abbreviation are to be written next to one another, separated with spaces, and in upper-case letters, even though this deviates from the conventional spelling (e.g. *A T V* for all-terrain vehicle instead of *ATV*).

[56] If this stipulation was followed correctly, the respondent was not penalised for the spelling error, but the occurrence was categorised as an acceptable protocol error. See 2.4.5.2.1 for a detailed description of the categories of errors that were used in this experiment.

- Do not make use of abbreviations if they are not heard in the audio recording, i.e. if the word "*etcetera*" is heard, transcribe it as "*etcetera*" and do not write "*etc.*".
- Acronyms are to be written entirely in upper-case letters, but not separated with spaces (e.g. *NASA*, *UNISA*).

Punctuation marks

- You are only allowed to use the following five punctuation marks: full stop, ellipsis, question mark, hyphen and comma.
- These punctuation marks are to be placed according to the rules of the written language. You may also use a hyphen to indicate partial words or to indicate words cut off at the beginning and end of utterances.
- Each utterance should end with one of the following three punctuation marks: full stop, ellipsis or question mark.

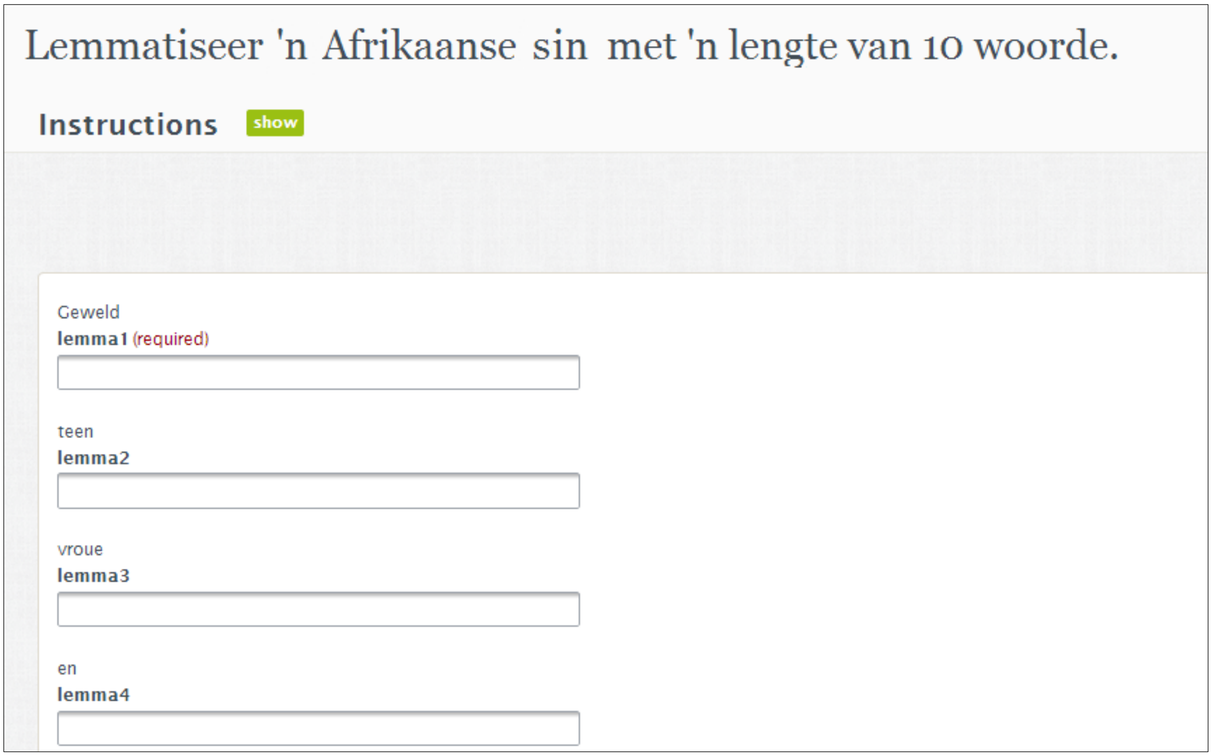
Annexure C

Description of software

C.1: *CrowdFlower*

CrowdFlower is a general-purpose crowd-sourcing application that allows customers to upload their own tasks to be carried out by users of labour channels such as *Amazon Mechanical Turk*, *TrialPay*, and *Samasource*. Small payments are paid per completed tasks, typically a few cents per task (http://en.wikipedia.org/wiki/*CrowdFlower*).

For the experiment in Chapter 2 (2.4) as well the experiment in Chapter 3 (3.4), we provided the tokenised list of words with an empty textbox underneath each word where the respondent could type the lemma. See Figure 19 for an example.



Lemmatiseer 'n Afrikaanse sin met 'n lengte van 10 woorde.

Geweld
lemma1 (required)

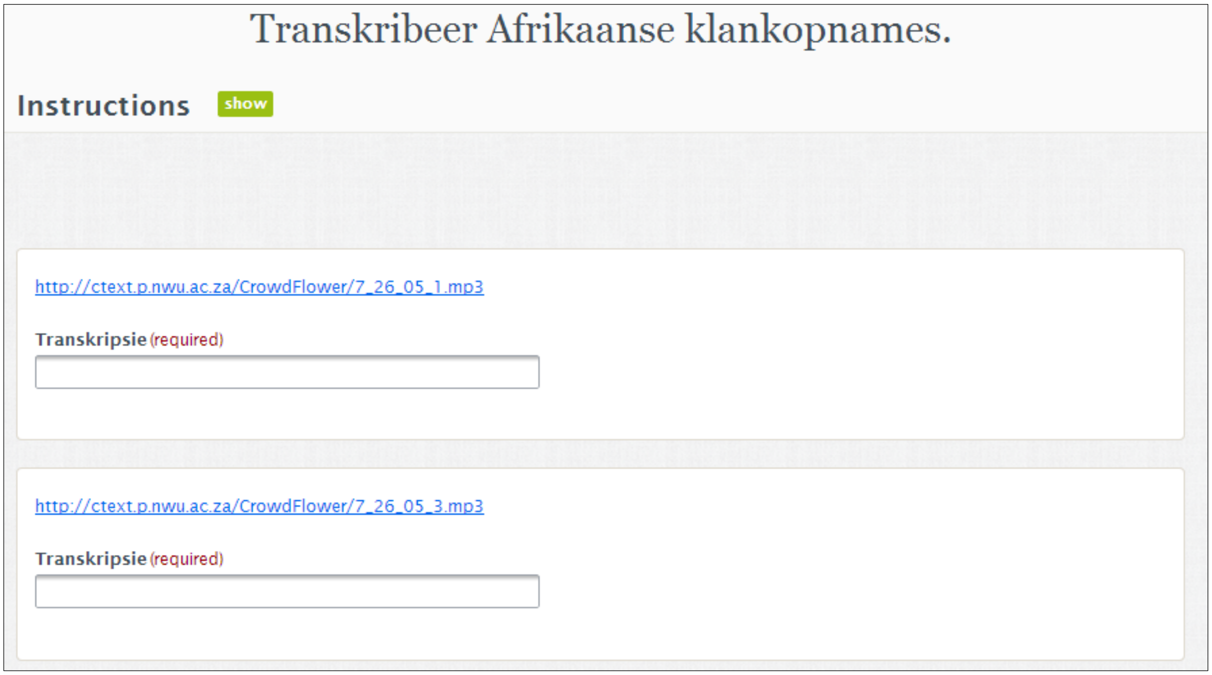
teen
lemma2

vroue
lemma3

en
lemma4

Figure 19: Example of lemmatisation in *CrowdFlower*

For the orthographic transcription in Chapter 2 (2.4) as well as in Chapter 3 (3.4), we provided the individual recordings (segmented on sentence level), and the respondent provided the transcription in the empty textbox underneath. See Figure 20 for an example.



Transkribeer Afrikaanse klankopnames.

Instructions

http://ctext.p.nwu.ac.za/CrowdFlower/7_26_05_1.mp3

Transkripsie (required)

http://ctext.p.nwu.ac.za/CrowdFlower/7_26_05_3.mp3

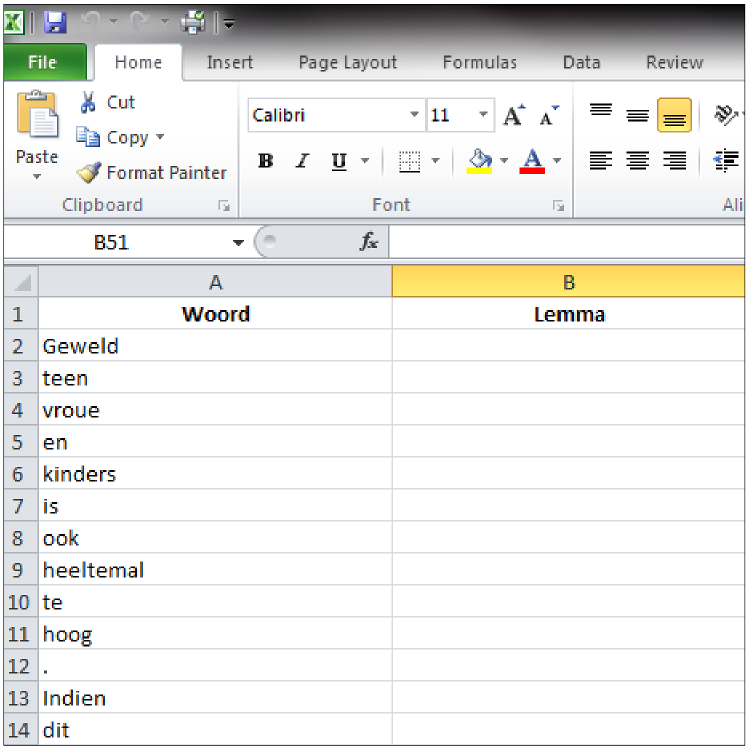
Transkripsie (required)

Figure 20: Example of orthographic transcription in *CrowdFlower*

C.2: *Microsoft Excel*

Microsoft Excel is a commercial spreadsheet application written and distributed by Microsoft for *Microsoft Windows* and *Mac OS X*. *Microsoft Excel* has the basic features of all spreadsheets, using a grid of cells arranged in numbered rows and letter-named columns to organise data manipulations like arithmetic operations (www.en.wikipedia.org/wiki/Microsoft_Excel).

This experiment provided the tokenised list of words in column A and the respondent had to type the lemma in column B. See Figure 21 for an example.



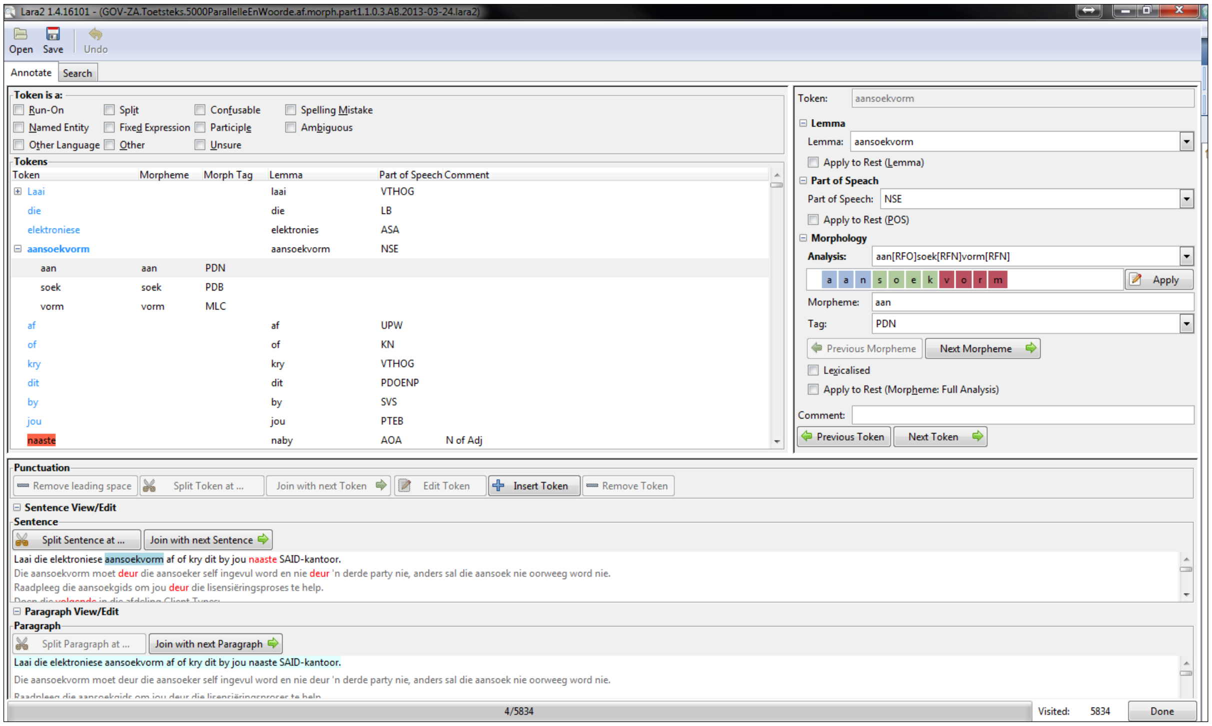
	A	B
1	**Woord**	**Lemma**
2	Geweld	
3	teen	
4	vroue	
5	en	
6	kinders	
7	is	
8	ook	
9	heeltemal	
10	te	
11	hoog	
12	.	
13	Indien	
14	dit	

Figure 21: Example of lemmatisation in *Excel*

C.3: *Lexicon Annotation and Regulation Assistant V2.0* (*LARA2*)

LARA2, developed by the Centre for Text Technology (CTexT), is domain-specific software for the annotation of tokens, lemmas, part-of-speech tags, and morphological analysis. The aim of this tool is to enable users who have limited or basic computer skills to develop annotated, machine-readable lexica. *LARA2* was used in the development of annotated lexica for ten (resource-scarce) South African languages for the South African *National Centre for Human Language Technologies* (*NCHLT*) project. The project aimed to annotate 50,000 tokens on four levels, i.e. tokenisation, lemmatisation, part-of-speech tagging and morphological analysis.

File management is automated, and a platform enabling the facilitation of lexicon annotation standardisation is provided. The tool also provides editing of token (token view), sentence (sentence view), and paragraph structure (paragraph view), and various attributes can be indicated by selecting the relevant text box.



Lara2 1.4.16101 - (GOV-ZA.Toetsteks.5000ParallelleEnWoorde.af.morph.part1.1.0.3.AB.2013-03-24.lara2)

Token is a:

□ Run-On □ Split □ Confusable □ Spelling Mistake
□ Named Entity □ Fixed Expression □ Participle □ Ambiguous
□ Other Language □ Other □ Unsure

Tokens

Token	Morpheme	Morph Tag	Lemma	Part of Speech	Comment
⊞ Laai			laai	VTHOG	
die			die	LB	
elektroniese			elektronies	ASA	
⊟ aansoekvorm			aansoekvorm	NSE	
aan	aan	PDN			
soek	soek	PDB			
vorm	vorm	MLC			
af			af	UPW	
of			of	KN	
kry			kry	VTHOG	
dit			dit	PDOENP	
by			by	SVS	
jou			jou	PTEB	
naaste			naby	AOA	N of Adj

Token: aansoekvorm

⊟ **Lemma**

Lemma: aansoekvorm

□ Apply to Rest (Lemma)

⊟ **Part of Speach**

Part of Speech: NSE

□ Apply to Rest (POS)

⊟ **Morphology**

Analysis: aan[RFO]soek[RFN]vorm[RFN]

a a n s o e k v o r m

Morpheme: aan

Tag: PDN

□ Lexicalised

□ Apply to Rest (Morpheme: Full Analysis)

Comment:

Punctuation

⊟ **Sentence View/Edit**

Sentence

Laai die elektroniese aansoekvorm af of kry dit by jou naaste SAID-kantoor.
Die aansoekvorm moet deur die aansoeker self ingevul word en nie deur 'n derde party nie, anders sal die aansoek nie oorweeg word nie.
Raadpleeg die aansoekgids om jou deur die lisensiëringsproses te help.
Doen die volgende in die afdeling Client Types:

⊟ **Paragraph View/Edit**

Paragraph

Laai die elektroniese aansoekvorm af of kry dit by jou naaste SAID-kantoor.
Die aansoekvorm moet deur die aansoeker self ingevul word en nie deur 'n derde party nie, anders sal die aansoek nie oorweeg word nie.
Raadpleeg die aansoekgids om jou deur die lisensiëringsproses te help.

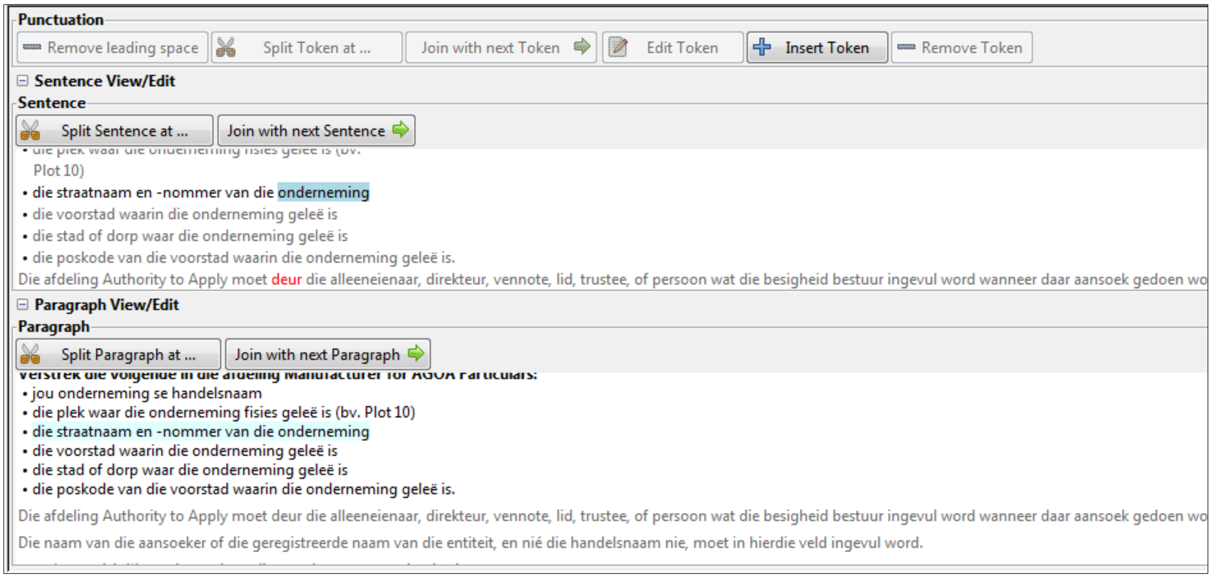
4/5834 Visited: 5834

Figure 22: Main window of *LARA2*

A **token view** (see Figure 22) shows one token per line with the lemma, morphemes, morpheme tags, part-of-speech tags and a comment field. Certain attributes can be selected for tokens, indicating whether the token is a confusable, named entity, other language, ambiguous, etc. Tokens can also be indicated as a spelling error, run-on or split, and the correction can be added. The punctuation in the text can also be edited by adding, removing or editing as necessary.

A **sentence view** (see Figure 23) displays the sentence format with the sentence containing the current token in bold. The next and previous sentences are displayed in light grey. Selected words from the token view are highlighted in the sentence. Tokens with the *confusable* attribute applied to them will be shown in red text colour. Sentence editing is enabled and users can split a sentence into multiple sentences, or join a sentence with another one.

In the **paragraph view** (see Figure 23) the text is displayed in paragraph format. The selected sentence highlighted in the sentence view will be highlighted in this paragraph. The next and previous paragraphs are displayed in light grey. A user can split the paragraph at any sentence or merge the current paragraph with the next.



- die plek waar die onderneming fisies geleë is (bv. Plot 10)
- die straatnaam en -nommer van die onderneming
- die voorstad waarin die onderneming geleë is
- die stad of dorp waar die onderneming geleë is
- die poskode van die voorstad waarin die onderneming geleë is.

Die afdeling Authority to Apply moet deur die alleeneienaar, direkteur, vennote, lid, trustee, of persoon wat die besigheid bestuur ingevul word wanneer daar aansoek gedoen wo

Verstrek die volgende in die afdeling Manufacturer for AGOA Particulars:

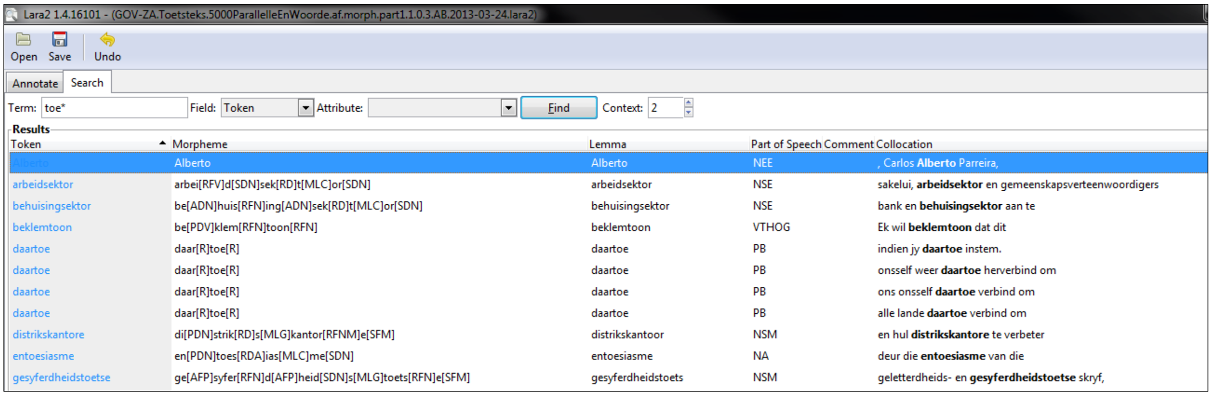
- jou onderneming se handelsnaam
- die plek waar die onderneming fisies geleë is (bv. Plot 10)
- die straatnaam en -nommer van die onderneming
- die voorstad waarin die onderneming geleë is
- die stad of dorp waar die onderneming geleë is
- die poskode van die voorstad waarin die onderneming geleë is.

Die afdeling Authority to Apply moet deur die alleeneienaar, direkteur, vennote, lid, trustee, of persoon wat die besigheid bestuur ingevul word wanneer daar aansoek gedoen wo

Die naam van die aansoeker of die geregistreerde naam van die entiteit, en nié die handelsnaam nie, moet in hierdie veld ingevul word.

Figure 23: Sentence and paragraph view in *LARA2*

The **search tab** (see Figure 24) can be used to find specific tokens by searching in the token, lemma, attribute or comment field (or combinations of these fields), and results are displayed with a variable context left and right.



Lara2 1.4.16101 - (GOV-ZA.Toetsteks.5000ParallelleEnWoorde.af.morph.part1.1.0.3.AB.2013-03-24.lara2)

Term: toe* Field: Token Attribute: Find Context: 2

Results

Token	Morpheme	Lemma	Part of Speech	Comment	Collocation
Alberto	Alberto	Alberto	NEE		, Carlos **Alberto** Parreira,
arbeidsektor	arbei[RFV]d[SDN]sek[RD]t[MLC]or[SDN]	arbeidsektor	NSE		sakelui, **arbeidsektor** en gemeenskapsverteenwoordigers
behuisingsektor	be[ADN]huis[RFN]ing[ADN]sek[RD]t[MLC]or[SDN]	behuisingsektor	NSE		bank en **behuisingsektor** aan te
beklemtoon	be[PDV]klem[RFN]toon[RFN]	beklemtoon	VTHOG		Ek wil **beklemtoon** dat dit
daartoe	daar[R]toe[R]	daartoe	PB		indien jy **daartoe** instem.
daartoe	daar[R]toe[R]	daartoe	PB		onsself weer **daartoe** herverbind om
daartoe	daar[R]toe[R]	daartoe	PB		ons onsself **daartoe** verbind om
daartoe	daar[R]toe[R]	daartoe	PB		alle lande **daartoe** verbind om
distrikskantore	di[PDN]strik[RD]s[MLG]kantor[RFNM]e[SFM]	distrikskantoor	NSM		en hul **distrikskantore** te verbeter
entoesiasme	en[PDN]toes[RDA]ias[MLC]me[SDN]	entoesiasme	NA		deur die **entoesiasme** van die
gesyferdheidstoetse	ge[AFP]syfer[RFN]d[AFP]heid[SDN]s[MLG]toets[RFN]e[SFM]	gesyferdheidstoets	NSM		geletterdheids- en **gesyferdheidstoetse** skryf,

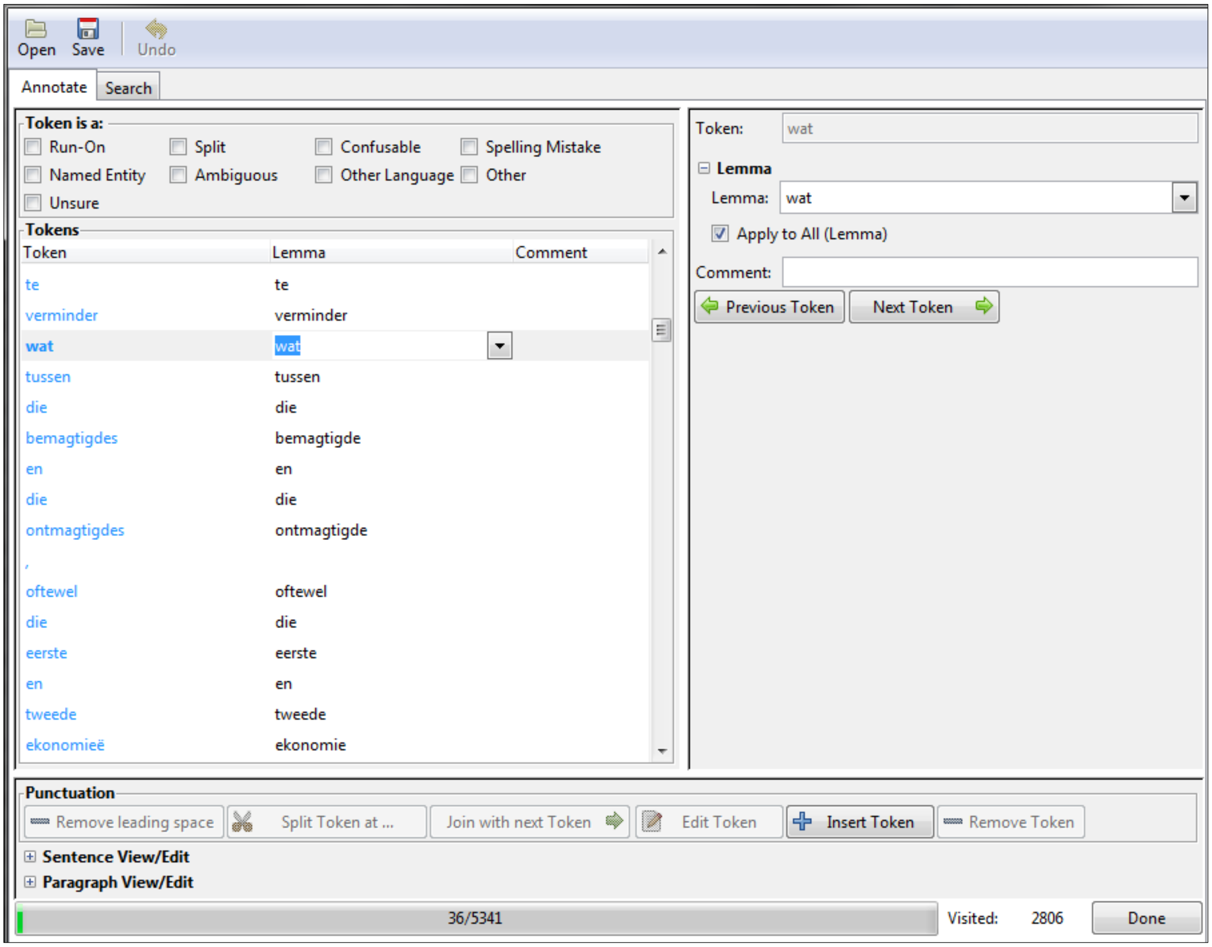
Figure 24: Search functionalities in *LARA2*

Users can undo the last ten actions, and *LARA2* also saves a backup copy every minute to ensure there is an up-to-date copy in case of a crash or power failure. *LARA2* uses an internal object model to represent standoff annotation, i.e. the text, with edge, token, sentence and paragraph layers are anchored to the character index in the text. No changes are made to the text itself; spelling errors or run-ons, etc. are annotated either in the edge or token layer. Originally XML was used to save the layers, but for speed reasons the file format was switched to a binary format using the storable module. The object model, however, still makes it easy to extract tokens/sentences/paragraphs or an XML representation if needed. *LARA2* was developed using Perl 5.10 and Gtk2-Perl with GTK+ 2.16, is distributed under an Open Source licence, and is available from http://www.nwu.ac.za/ctext.

For the experiment in Chapter 3, two versions of *LARA2* were used. A scaled-down version that included no additional features relevant to the task of lemmatisation was developed (referred to as *LARALite* in this study). A full version (*LARAFull*) was developed by customising *LARA2*.

C.3.1: *LARALite*

LARALite (see Figure 25) was used as an example of domain-specific software and to compare *LARALite* with *CrowdFlower* and *Microsoft Excel* to confirm that the basic interface has no direct effect on the task. For the experiment in Chapter 3, respondents only used the token view and did not need to indicate any additional attributes or correct spelling and tokenisation errors as the data was corrected beforehand. The respondents' only task was to provide the lemmas in the lemma column.



Token is a:

□ Run-On □ Split □ Confusable □ Spelling Mistake
□ Named Entity □ Ambiguous □ Other Language □ Other
□ Unsure

Tokens

Token	Lemma	Comment
te	te	
verminder	verminder	
wat	wat	
tussen	tussen	
die	die	
bemagtigdes	bemagtigde	
en	en	
die	die	
ontmagtigdes	ontmagtigde	
,		
oftewel	oftewel	
die	die	
eerste	eerste	
en	en	
tweede	tweede	
ekonomieë	ekonomie	

Token: wat

Lemma

Lemma: wat

☑ Apply to All (Lemma)

Comment:

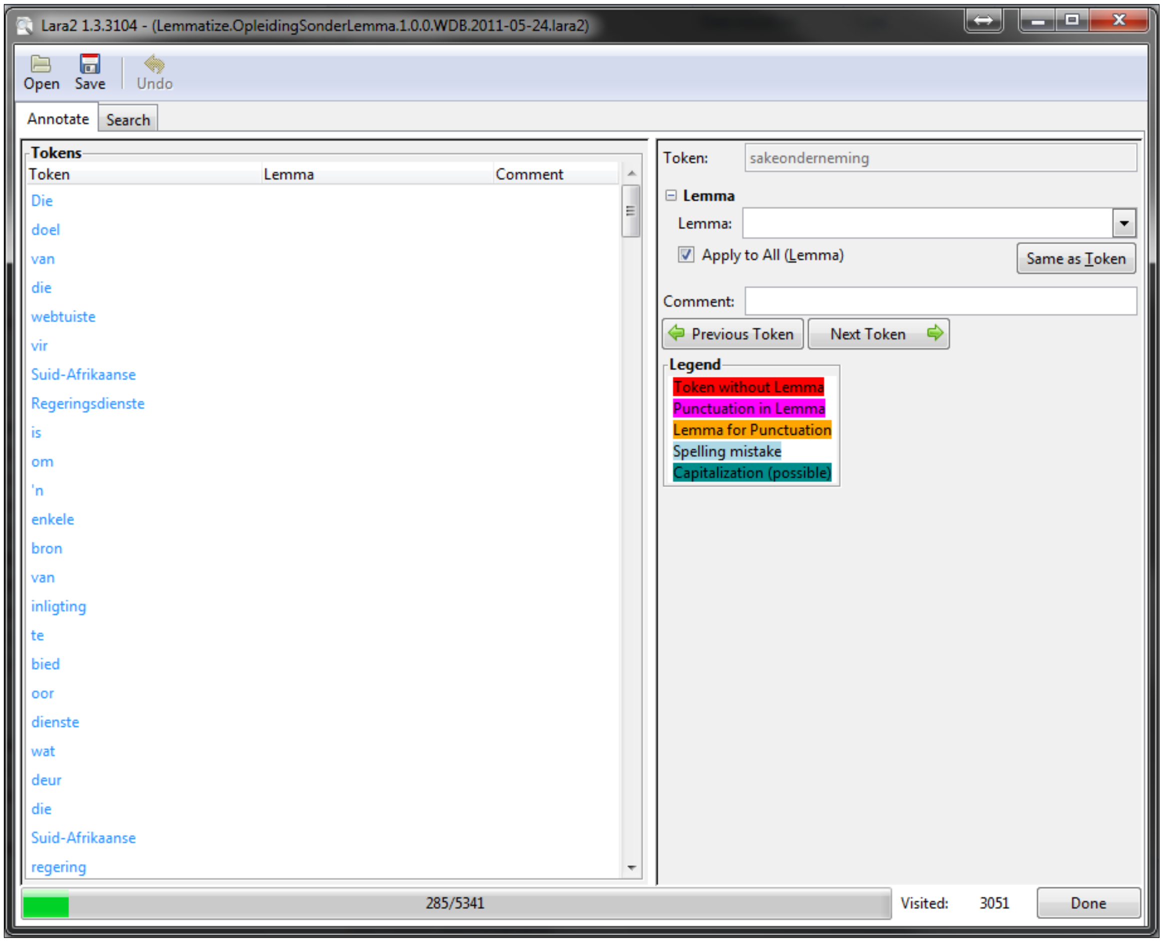
36/5341

Visited: 2806

Figure 25: Main window of *LARALite*

C.3.2: *LARAFull*

To customise *LARA2*, we removed additional features, such as the sentence view and POS tagging etc. Features relevant to the task, i.e. features aimed at improving the accuracy of the annotations, and aimed at reducing the time needed to complete the annotations (see description below), were added. This resulted in tailor-made software, specifically aimed at the task of lemmatisation.


Lara2 1.3.3104 - (Lemmatize.OpleidingSonderLemma.1.0.0.WDB.2011-05-24.lara2)
Open
Save
Undo
Annotate
Search
Tokens
Previous Token
Next Token
Done

Token	Lemma	Comment
Die		
doel		
van		
die		
webtuiste		
vir		
Suid-Afrikaanse		
Regeringsdienste		
is		
om		
'n		
enkele		
bron		
van		
inligting		
te		
bied		
oor		
dienste		
wat		
deur		
die		
Suid-Afrikaanse		
regering		

Token: sakeonderneming

⊟ Lemma

Lemma:

☑ Apply to All (Lemma) Same as Token

Comment:

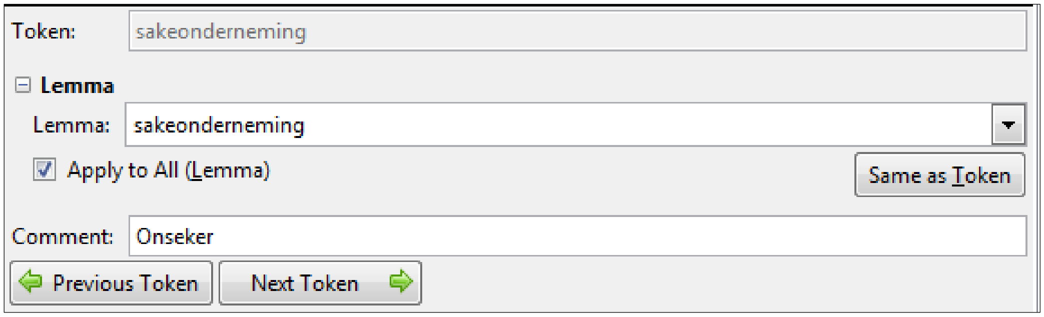
Legend
- Token without Lemma
- Punctuation in Lemma
- Lemma for Punctuation
- Spelling mistake
- Capitalization (possible)

285/5341 Visited: 3051

Figure 26: Main window of *LARAFull*

LARAFull (see Figure 26) has six features relevant to the task of lemmatisation:

1. An “Apply to All” checkbox that automatically populates the lemma for all subsequent occurrences of the specific word if the user activates the feature;
2. A button (“Same as Token”) that automatically populates the current lemma field with the same string that appears in the token field (see Figure 27);



Token: sakeonderneming

⊟ Lemma

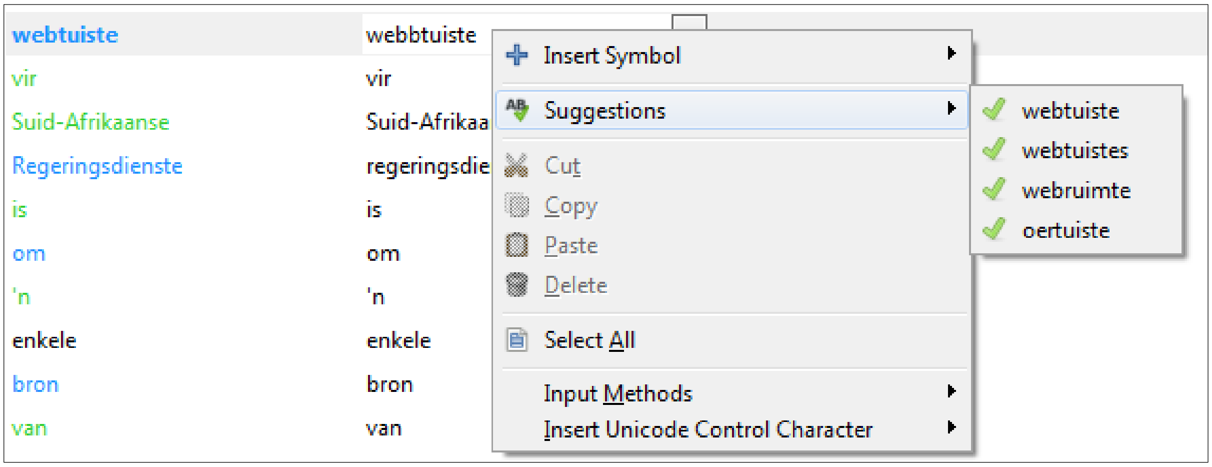
Lemma: sakeonderneming

☑ Apply to All (Lemma)

Comment: Onseker

Figure 27: “Apply to All” and “Same as Token” features of *LARAFull*

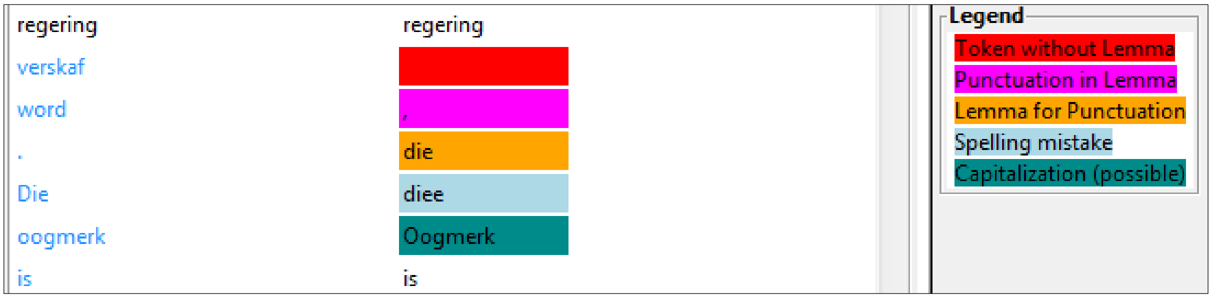
3. A spelling checker that flags any spelling errors made by the respondent. Suggestions are also provided in a pop-up window (see Figure 28);



webtuiste	webbtuiste
vir	vir
Suid-Afrikaanse	Suid-Afrikaa
Regeringsdienste	regeringsdie
is	is
om	om
'n	'n
enkele	enkele
bron	bron
van	van

Figure 28: Spelling checking and suggestion features of *LARAFull*

4. Capitalised lemmas are flagged as capitalisation errors if the lemma appears in the lowercase part of the spelling checker lexicon;
5. Empty lemma fields are flagged if a user skips a required entry; and
6. Punctuation is flagged in the lemma entry if the punctuation differs from punctuation in the token field (see Figure 29).



regering	regering
verskaf	🟥
word	,
.	die
Die	diee
oogmerk	Oogmerk
is	is

Figure 29: Automatic flags in *LARAFull*

C.4: *Praat*

Praat (the Dutch word for “talk”; see Figure 30) is a free scientific software programme for the analysis of speech in phonetics. It has been designed and is being continuously developed by Paul Boersma and David Weenink of the University of Amsterdam. It can run on a wide range of operating systems, including various *UNIX* versions, *Mac* and *Microsoft Windows*. The programme also supports speech synthesis while articulatory synthesis and transcriptions can be added in separate textgrid files (http://en.wikipedia.org/wiki/*Praat*).

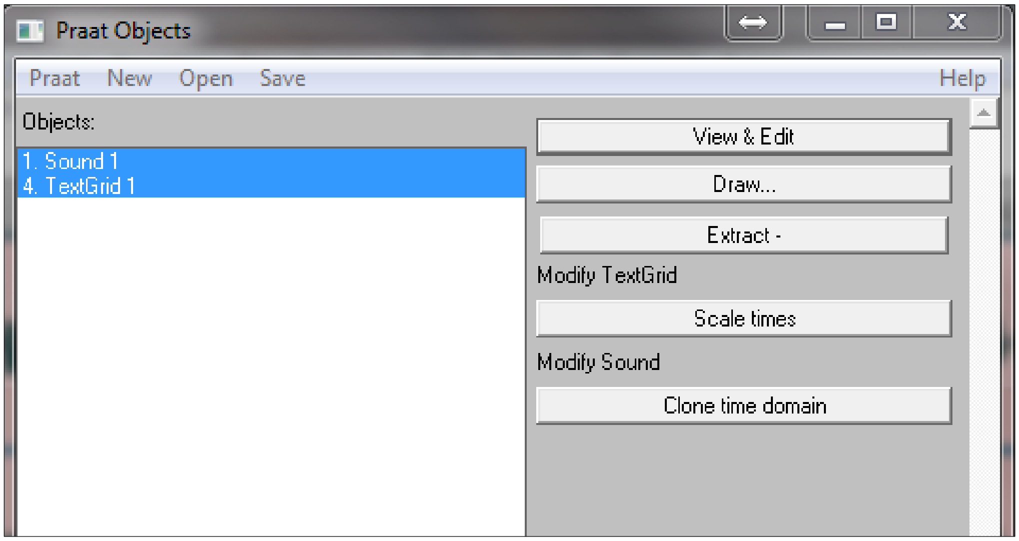
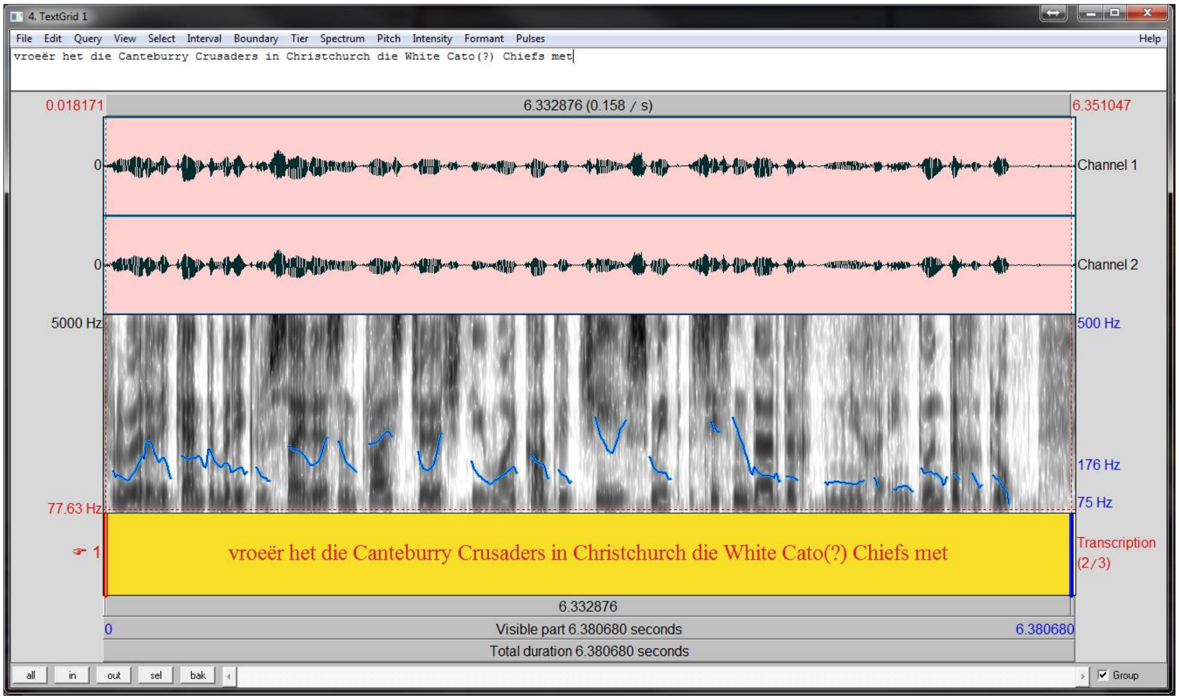


Figure 30: File window in *Praat*

In *Praat*, transcriptions are saved in associated *.textgrid* files. A user selects the sound and the relevant *.textgrid* file and then clicks on “View &Edit”. This opens a new window where a waveform and spectrogram is displayed. Transcriptions are displayed in a textbox above and directly underneath the waveform (see Figure 31).



vroeër het die Canteburry Crusaders in Christchurch die White Cato(?) Chiefs met

vroeër het die Canteburry Crusaders in Christchurch die White Cato(?) Chiefs met

Figure 31: Main window of *Praat*

A user has the options to zoom into shorter segments of the waveform and observe acoustic properties such as periodicity, energy, formants, etc. Portions of the waveform can also be played by clicking on the bar directly above the selection, or by using a keyboard shortcut (see Figure 32).

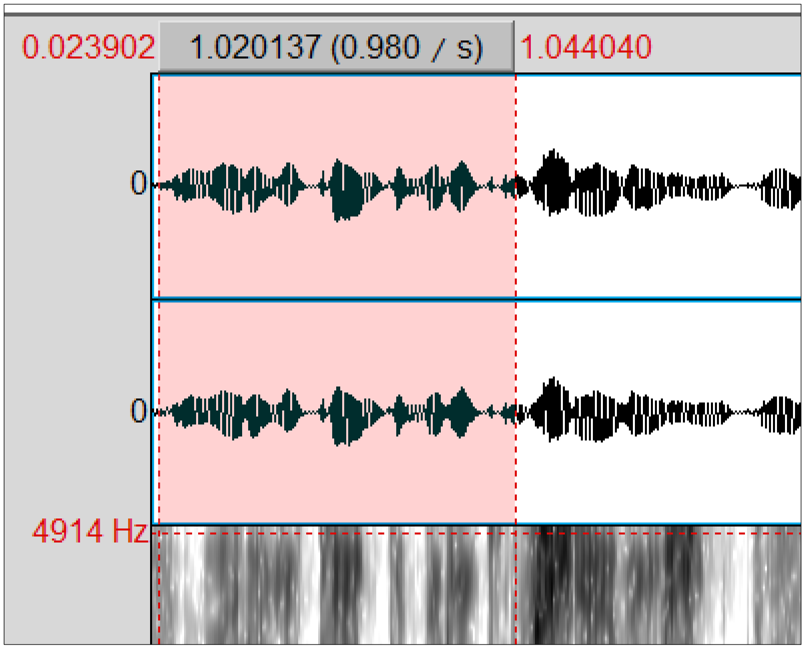
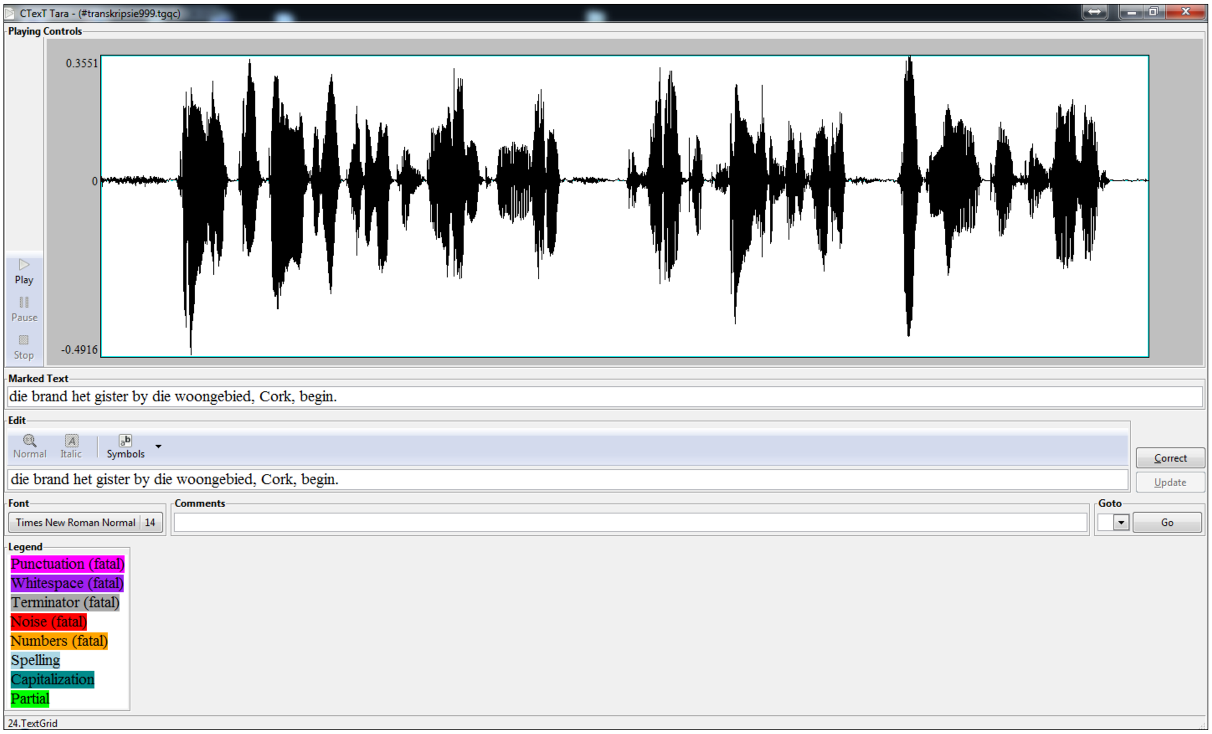


Figure 32: Play controls and graph in *Praat*

C.5: *TARA*

TARA (see Figure 33) is an audio data transcription environment developed for performing orthographic transcriptions of speech data. The aim of *TARA* is to enable the annotators to focus on the task of transcribing the data, by automatically performing basic functionalities such as opening and saving files and protocol checking. A graph depicting the sound is visible at the top of the screen and the view includes play/pause controls which can be toggled if the transcriber wishes to replay a particular data section. The window displays two versions of the transcription, the original and new, for purposes of quality control (see Figure 34).


CTexT Tara - (#transkripsie999.tgqc)

Marked Text: die brand het gister by die woongebied, Cork, begin.

Edit: die brand het gister by die woongebied, Cork, begin.

Font: Times New Roman Normal 14

Legend:
- Punctuation (fatal)
- Whitespace (fatal)
- Terminator (fatal)
- Noise (fatal)
- Numbers (fatal)
- Spelling
- Capitalization
- Partial

24.TextGrid

Figure 33: Main window of *TARA*

Each audio file is segmented on utterance level, one sentence per utterance, and stored as separate *.wav* and *.textgrid* files. These audio files are automatically loaded as a user completes a sentence. An interactive graph is created for each utterance. The user can select a section of the graph only to listen to a portion of the file. Keyboard shortcuts are available for playback controls. Visual representation is also beneficial for identifying noises.

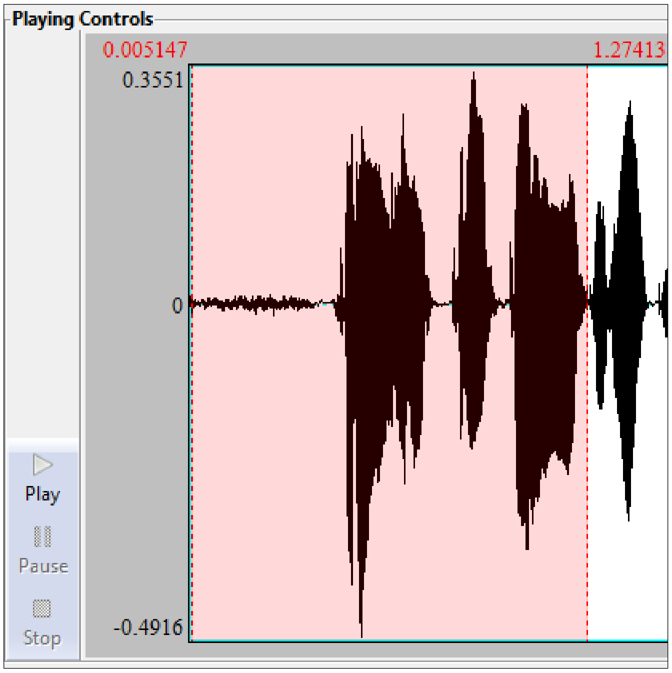
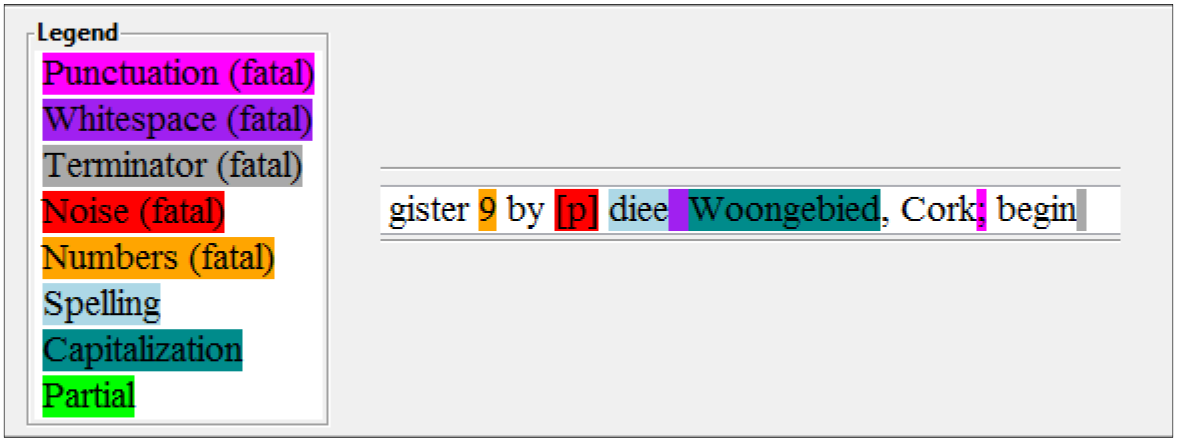


Figure 34: Play controls and graph in *TARA*

TARA performs automatic protocol checking and indicates spelling errors, invalid spaces or punctuation, as well as incorrect capitalisation. The transcriber can only continue with the transcription of a subsequent segment once all fatal errors, such as incorrect usage of punctuation marks, have been corrected (see Figure 35).



gister 9 by [p] diee Woongebied, Cork; begin

Figure 35: Automatic flags in *TARA*

If a word is flagged in the “Marked Text” section, a user can right click on the word in the “Edit” section and select “Suggestions” for a list of possible suggestions for the word.

If a word is flagged in the “Marked Text” section, but a user is sure that it is spelled correctly, he/she can right click on the word in the “Edit” section and select “Add to Dictionary”. The word will not be flagged in any future recordings (see Figure 36).

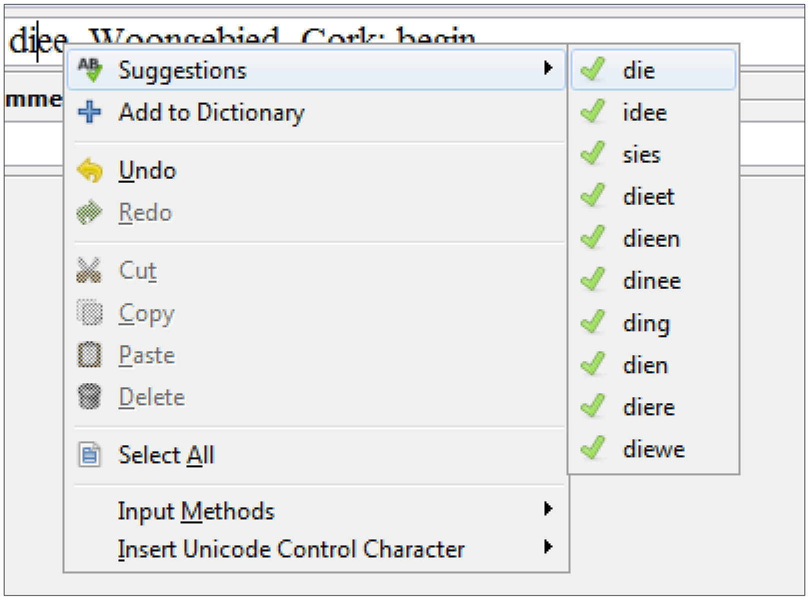


Figure 36: Spelling checking and suggestions in *TARA*

TARA was developed using Perl 5.10 and Gtk2-Perl with GTK+ 2.16. *TARA* uses a XML project file that lists the language, annotation protocol used and files that are in the project. Additionally it uses a XML file to store extra information (date, speaker, comments, etc.) for each utterance. This XML is stored in a time-stamped directory (based on the start of the current session) of the *TARA* meta-directory as well as a backup of the utterance before changes were made. This is to facilitate both tracking of changes and data redundancy. *TARA* is distributed under an Open Source licence and is available from http://www.nwu.ac.za/ctext.

Annexure D

Tables from Chapter 2

	Expert	Novice		Laymen	
	Value	Mean	Standard deviation	Mean	Standard deviation
Time (s)	3874.000	3862.500	346.980	3779.200	625.480
Precision	0.948	0.304	0.111	0.363	0.075
Recall	0.807	0.506	0.125	0.507	0.123
F-Score	0.872	0.377	0.121	0.413	0.082
Accuracy	0.971	0.756	0.044	0.781	0.041
Capitalisation errors	0.000	32.400	25.713	46.400	24.708
Spelling errors	2.000	22.500	10.266	24.900	17.540
Empty responses	0.000	1.500	2.321	1.700	2.541

Table 40: Mean values and standard deviation of variables in Task A, Chapter 2

	Expert	Novices		Laymen	
	Value	Mean	Standard deviation	Mean	Standard deviation
Time (s)	4058.000	3938.500	612.4498	4688.500	1083.502
Total Errors	29.000	267.400	131.2201	162.800	81.887
Transcription errors					
Insertions	0.000	12.000	13.5319	3.200	3.011
Deletions	0.000	41.700	62.9886	6.300	3.129
Substitutions	1.000	19.300	18.9036	7.800	3.360
Transpositions	0.000	0.200	0.4216	0.000	0.000
Language errors					
Spelling error	6.000	69.500	36.2683	46.900	45.759
Capitalisation	3.000	21.000	7.7603	18.700	4.877
Punctuation	14.000	23.900	7.6659	22.000	7.364
Hyphen	1.000	8.000	4.4969	10.700	5.034
Compound	3.000	18.800	9.9421	18.500	16.154
Protocol errors					
Capitalisation	0.000	11.700	15.1588	6.200	9.295
Number	0.000	4.300	4.5717	4.400	9.180
Abbreviation	0.000	4.700	2.5841	3.500	2.369
Acronym	0.000	1.200	0.9189	1.600	1.075
Punctuation	0.000	7.400	7.3515	6.200	7.239
White space	1.000	13.400	10.0797	4.400	4.502
Terminator	0.000	10.300	14.4226	2.400	3.806

Table 41: Mean values and standard deviation of variables in Task B, Chapter 2

Annexure E

Phonemes in ASR systems

@

@i

@u

A_c

E

N

O

S

Z

a

b

d

eu_c

f

g

h_b

i

i@

j

k

l

m

n

p

r

s

sil

t

u

u@

u_

u_y

v

w

x

y

z

{

Annexure F

Comparison of systems containing only one category of errors

	Language Errors		Transcription errors	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Expert	59.881	2.975	59.252	2.862
Trained novice	60.663	2.989	59.645	3.206
Untrained novice	62.751	2.661	62.941	2.951

Table 42: WER of systems containing one category of errors

Annexure G

Tables from Chapter 4, Experiment 1, Task B

Increment	Gold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	103.34	104.35	105.07	107.78
2 (20%)	91.37	92.05	92.78	97.17
3 (30%)	79.64	80.25	81.28	86.16
4 (40%)	73.73	74.18	75.51	81.02
5 (50%)	70.11	70.67	71.37	76.65
6 (60%)	67.51	67.75	69.34	73.54
7 (70%)	65.56	66.19	67.24	71.99
8 (80%)	61.93	62.33	63.62	68.86
9 (90%)	60.32	60.89	62.05	67.26
10 (100%)	59.08	59.68	60.56	65.97

Table 43: WER of systems

Increment	Gold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	28.67	28.87	29.13	30.05
2 (20%)	24.60	24.80	24.99	25.97
3 (30%)	22.25	22.42	22.67	23.53
4 (40%)	21.04	21.19	21.35	22.30
5 (50%)	20.17	20.24	20.27	21.14
6 (60%)	19.31	19.38	19.57	20.47
7 (70%)	18.40	18.44	18.79	19.47
8 (80%)	17.83	17.92	18.15	19.08
9 (90%)	17.23	17.38	17.67	18.67
10 (100%)	16.92	17.07	17.33	18.35

Table 44: PER of systems

Increment	Gold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	54.19	53.73	52.98	50.34
2 (20%)	60.75	60.52	59.75	56.43
3 (30%)	65.44	64.98	64.30	61.36
4 (40%)	67.76	67.23	66.67	63.58
5 (50%)	69.40	69.07	68.69	65.49
6 (60%)	70.81	70.64	69.75	66.98
7 (70%)	72.05	71.59	70.87	67.84
8 (80%)	73.15	72.76	72.17	68.90
9 (90%)	73.86	73.56	72.80	69.91
10 (100%)	74.57	74.31	73.58	70.45

Table 45: Recall of systems (word level)

Increment	Gold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	80.50	80.36	79.93	78.28
2 (20%)	84.25	84.08	83.87	82.52
3 (30%)	86.50	86.42	86.16	85.07
4 (40%)	87.65	87.55	87.37	86.20
5 (50%)	88.40	88.38	88.28	87.30
6 (60%)	89.17	89.15	88.93	87.96
7 (70%)	89.82	89.72	89.52	88.70
8 (80%)	90.25	90.21	90.01	89.23
9 (90%)	90.64	90.64	90.44	89.54
10 (100%)	90.97	90.87	90.71	89.86

Table 46: Recall of systems (phone level)

Increment	Gold		Expert		Trained novice		Untrained novice		*P*-value
	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation	Repeated measures ANOVA (Greenhouse-Geisser corrected)
	95% Confidence interval		95% Confidence interval		95% Confidence interval		95% Confidence interval		
	Upper	Lower	Upper	Lower	Upper	Lower	Upper	Lower	
1	103.34	2.40	104.35	1.87	105.07	2.46	107.78	2.45	< 0.001
	104.83	101.86	105.51	103.19	106.59	103.55	109.30	106.27	
2	91.37	4.54	92.05	4.39	92.78	4.82	97.17	4.93	< 0.001
	94.18	88.55	94.77	89.33	95.77	89.80	100.22	94.11	
3	79.64	2.69	80.25	2.18	81.28	2.68	86.16	2.93	< 0.001
	81.31	77.98	81.60	78.90	82.94	79.61	87.97	84.34	
4	73.73	2.68	74.18	2.58	75.51	1.85	81.02	1.82	< 0.001
	75.40	72.07	75.78	72.58	76.66	74.37	82.14	79.89	
5	70.11	3.33	70.67	3.25	71.37	2.56	76.65	2.33	< 0.001
	72.17	68.05	72.69	68.66	72.96	69.79	78.10	75.21	
6	67.51	4.50	67.75	4.33	69.34	4.38	73.54	4.01	< 0.001
	70.30	64.72	70.43	65.07	72.05	66.62	76.02	71.05	
7	65.56	2.39	66.19	2.23	67.24	2.34	71.99	2.39	< 0.001
	67.04	64.08	67.57	64.80	68.69	65.79	73.47	70.51	
8	61.93	3.99	62.33	4.06	63.62	3.95	68.86	4.16	< 0.001
	64.40	59.45	64.85	59.82	66.07	61.17	71.44	66.29	
9	60.32	2.77	60.89	2.74	62.05	3.09	67.26	2.76	< 0.001
	62.04	58.61	62.59	59.20	63.97	60.14	68.97	65.56	
10	59.08	3.03	59.68	2.73	60.56	2.71	65.97	2.56	< 0.001
	60.96	57.21	61.37	57.99	62.24	58.88	67.56	64.38	

Table 47: Means, standard deviations and 95% confidence levels of WER

Increment	**Expert**			**Trained novice**			**Untrained novice**		
	Mean difference	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**	**Mean difference**	**Bonferroni corrected *p***	**Effect size (d)**
1	1.009	0.437	-0.494	1.727	0.021	-0.750	4.438	< 0.001	-1.930
2	0.682	1.000	-0.161	1.417	0.047	-0.319	5.800	< 0.001	-1.290
3	0.607	1.000	-0.261	1.634	0.013	-0.641	6.513	< 0.001	-2.441
4	0.448	1.000	-0.179	1.781	< 0.001	-0.814	7.285	< 0.001	-3.351
5	0.564	1.000	-0.181	1.264	0.064	-0.449	6.546	< 0.001	-2.402
6	0.240	1.000	-0.057	1.826	< 0.001	-0.433	6.028	< 0.001	-1.491
7	0.626	0.785	-0.285	1.677	0.002	-0.748	6.428	< 0.001	-2.839
8	0.407	1.000	-0.107	1.693	0.002	-0.449	6.936	< 0.001	-1.794
9	0.569	1.000	-0.218	1.730	0.019	-0.622	6.938	< 0.001	-2.648
10	0.593	0.542	-0.217	1.474	0.001	-0.541	6.884	< 0.001	-2.589

Table 48: Mean difference, *p*-values and *d*-values of all increments of WER of systems compared to Gold

Increment	Gold		Expert		Trained novice		Untrained novice		*P*-value
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Repeated measures ANOVA (Greenhouse-Geisser corrected)**
	95% Confidence interval		**95% Confidence interval**		**95% Confidence interval**		**95% Confidence interval**		
	Upper	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	
1	54.187	1.329	53.730	1.425	52.981	1.579	50.335	1.523	< 0.001
	55.011	53.363	54.613	52.847	53.960	52.002	51.279	49.391	
2	60.748	1.478	60.519	1.463	59.747	1.605	56.433	1.696	< 0.001
	61.664	59.832	61.426	59.612	60.742	58.752	57.484	55.382	
3	65.437	0.896	64.976	1.124	64.304	1.197	61.359	1.022	< 0.001
	65.992	64.882	65.672	64.280	65.046	63.562	61.992	60.726	
4	67.763	1.149	67.231	0.999	66.673	1.006	63.584	1.055	< 0.001
	68.475	67.051	67.850	66.612	67.296	66.050	64.238	62.930	
5	69.400	1.197	69.073	1.293	68.687	1.191	65.494	0.887	< 0.001
	70.142	68.658	69.874	68.272	69.425	67.949	66.044	64.944	
6	70.812	1.640	70.640	1.618	69.746	1.688	66.981	1.582	< 0.001
	71.828	69.796	71.643	69.637	70.792	68.700	67.961	66.001	
7	72.046	1.106	71.585	1.049	70.872	1.074	67.839	0.620	< 0.001
	72.732	71.360	72.235	70.935	71.538	70.206	68.223	67.455	
8	73.145	1.607	72.762	1.610	72.167	1.476	68.898	1.445	< 0.001
	74.141	72.149	73.760	71.764	73.082	71.252	69.793	68.003	
9	73.857	1.225	73.559	1.088	72.795	1.103	69.909	1.150	< 0.001
	74.616	73.098	74.234	72.884	73.479	72.111	70.622	69.196	
10	74.573	1.235	74.311	1.073	73.576	1.205	70.453	1.252	< 0.001
	75.338	73.808	74.976	73.646	74.323	72.829	71.229	69.677	

Table 49: Means, standard deviations and 95% confidence levels of recall (word level)

Increment	Expert			Trained novice			Untrained novice		
	Mean difference	Bonferroni corrected *p*	Effect size (d)	Mean difference	Bonferroni corrected *p*	Effect size (d)	Mean difference	Bonferroni corrected *p*	Effect size (d)
1	0.457	0.211	0.350	1.206	< 0.001	0.871	3.852	< 0.001	2.841
2	0.229	1.000	0.164	1.001	0.004	0.684	4.315	< 0.001	2.859
3	0.461	0.093	0.478	1.133	< 0.001	1.130	4.078	< 0.001	4.473
4	0.532	0.013	0.521	1.090	< 0.001	1.064	4.179	< 0.001	3.994
5	0.327	0.356	0.277	0.713	0.001	0.629	3.906	< 0.001	3.908
6	0.172	1.000	0.111	1.066	< 0.001	0.675	3.831	< 0.001	2.506
7	0.461	0.240	0.451	1.174	< 0.001	1.135	4.207	< 0.001	4.946
8	0.383	0.159	0.251	0.978	< 0.001	0.668	4.247	< 0.001	2.930
9	0.298	0.635	0.271	1.062	< 0.001	0.960	3.948	< 0.001	3.503
10	0.262	0.588	0.239	0.997	< 0.001	0.861	4.120	< 0.001	3.492

Table 50: Mean difference, *p*-value and *d*-value of all increments of recall of systems compared to Gold (word level)

Annexure H

Tables from Chapter 4, Experiment 2, Task B

Increment	HGold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	115.93	104.35	105.07	107.78
2 (20%)	105.48	92.05	92.78	97.17
3 (30%)	94.84	80.25	81.28	86.16
4 (40%)	88.24	74.18	75.51	81.02
5 (50%)	84.89	70.67	71.37	76.65
6 (60%)	79.95	67.75	69.34	73.54
7 (70%)	78.08	66.19	67.24	71.99
8 (80%)	74.84	62.33	63.62	68.86
9 (90%)	73.31	60.89	62.05	67.26
10 (100%)	70.93	59.68	60.56	65.97

Table 51: WER of systems

Increment	HGold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	32.77	28.87	29.13	30.05
2 (20%)	29.01	24.80	24.99	25.97
3 (30%)	26.50	22.42	22.67	23.53
4 (40%)	24.59	21.19	21.35	22.30
5 (50%)	23.42	20.24	20.27	21.14
6 (60%)	22.35	19.38	19.57	20.47
7 (70%)	21.79	18.44	18.79	19.47
8 (80%)	21.05	17.92	18.15	19.08
9 (90%)	20.59	17.38	17.67	18.67
10 (100%)	20.12	17.07	17.33	18.35

Table 52: PER of systems

Increment	HGold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	45.72	53.73	52.98	50.34
2 (20%)	53.25	60.52	59.75	56.43
3 (30%)	58.63	64.98	64.30	61.36
4 (40%)	61.59	67.23	66.67	63.58
5 (50%)	63.34	69.07	68.69	65.49
6 (60%)	65.23	70.64	69.75	66.98
7 (70%)	66.43	71.59	70.87	67.84
8 (80%)	67.52	72.76	72.17	68.90
9 (90%)	68.57	73.56	72.80	69.91
10 (100%)	69.69	74.31	73.58	70.45

Table 53: Recall of systems (word level)

Increment	HGold (0% errors)	Expert (1.09% errors)	Trained Novice (4.05% errors)	Untrained Novice (17.59% errors)
1 (10%)	75.88	80.36	79.93	78.28
2 (20%)	80.13	84.08	83.87	82.52
3 (30%)	82.76	86.42	86.16	85.07
4 (40%)	84.32	87.55	87.37	86.20
5 (50%)	85.44	88.38	88.28	87.30
6 (60%)	86.45	89.15	88.93	87.96
7 (70%)	87.10	89.72	89.52	88.70
8 (80%)	87.61	90.21	90.01	89.23
9 (90%)	88.08	90.64	90.44	89.54
10 (100%)	88.57	90.87	90.71	89.86

Table 54: Recall of systems (phone level)

Increment	HGold		Expert		Trained novice		Untrained novice		*P*-value
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Repeated measures ANOVA (Greenhouse-Geisser corrected)**
	95% Confidence interval		**95% Confidence interval**		**95% Confidence interval**		**95% Confidence interval**		
	Upper	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	**Upper**	**Lower**	
1	115.931	2.267	104.353	1.874	105.071	2.455	107.782	2.445	< 0.001
	117.336	114.526	105.514	103.191	106.593	103.549	109.297	106.266	
2	105.484	4.592	92.049	4.392	92.784	4.817	97.167	4.927	< 0.001
	108.33	102.638	94.771	89.327	95.77	89.799	100.221	94.114	
3	94.841	2.391	80.251	2.177	81.278	2.684	86.157	2.929	< 0.001
	96.323	93.359	81.601	78.902	82.941	79.614	87.972	84.342	
4	88.243	2.67	74.18	2.577	75.513	1.851	81.017	1.817	< 0.001
	89.899	86.588	75.778	72.583	76.66	74.365	82.143	79.891	
5	84.887	3.375	70.673	3.249	71.373	2.557	76.655	2.33	< 0.001
	86.979	82.795	72.686	68.659	72.957	69.788	78.099	75.21	
6	79.955	4.142	67.75	4.329	69.336	4.382	73.538	4.011	< 0.001
	82.522	77.387	70.433	65.067	72.052	66.62	76.023	71.052	
7	78.081	4.255	66.187	2.234	67.238	2.336	71.989	2.386	< 0.001
	80.718	75.443	67.572	64.802	68.686	65.79	73.468	70.51	
8	74.836	3.907	62.335	4.057	63.621	3.951	68.864	4.16	< 0.001
	77.257	72.414	64.849	59.82	66.07	61.172	71.442	66.286	
9	73.311	2.96	60.894	2.736	62.055	3.087	67.263	2.755	< 0.001
	75.146	71.477	62.59	59.198	63.968	60.141	68.971	65.555	
10	70.929	3.725	59.677	2.725	60.558	2.711	65.968	2.561	< 0.001
	73.238	68.621	61.367	57.988	62.238	58.878	67.556	64.381	

Table 55: Means, standard deviations and 95% confidence levels for WER

	Expert			Trained novice			Untrained novice		
Increment	Mean difference	Bonferroni corrected *p*	Effect size (d)	Mean difference	Bonferroni corrected *p*	Effect size (d)	Mean difference	Bonferroni corrected *p*	Effect size (d)
1	11.581	< 0.001	5.868	10.861	< 0.001	4.845	8.151	< 0.001	3.643
2	13.434	< 0.001	3.152	12.704	< 0.001	2.845	8.314	< 0.001	1.841
3	14.591	< 0.001	6.726	13.561	< 0.001	5.625	8.681	< 0.001	3.424
4	14.063	< 0.001	5.649	12.733	< 0.001	5.841	7.223	< 0.001	3.335
5	14.217	< 0.001	4.523	13.517	< 0.001	4.758	8.237	< 0.001	2.992
6	12.205	< 0.001	3.037	10.615	< 0.001	2.625	6.415	< 0.001	1.659
7	11.891	< 0.001	3.689	10.841	< 0.001	3.330	6.091	< 0.001	1.862
8	12.506	< 0.001	3.309	11.216	< 0.001	3.009	5.976	< 0.001	1.560
9	12.421	< 0.001	4.592	11.261	< 0.001	3.923	6.051	< 0.001	2.230
10	11.249	< 0.001	3.634	10.369	< 0.001	3.356	4.959	< 0.001	1.636

Table 56: Mean difference and *p*-value of all increments of WER of systems compared to HGold

Increment	HGold		Expert		Trained novice		Untrained novice		*P*-value
	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation	Mean	Standard Deviation	Repeated measures ANOVA (Greenhouse-Geisser corrected)
	95% Confidence interval		95% Confidence interval		95% Confidence interval		95% Confidence interval		
	Upper	Lower	Upper	Lower	Upper	Lower	Upper	Lower	
1	45.721	1.398	53.730	1.425	52.981	1.579	50.335	1.523	< 0.001
	46.588	44.854	54.613	52.847	53.960	52.002	51.279	49.391	
2	53.246	1.616	60.519	1.463	59.747	1.605	56.433	1.696	< 0.001
	54.248	52.244	61.426	59.612	60.742	58.752	57.484	55.382	
3	58.634	1.409	64.976	1.124	64.304	1.197	61.359	1.022	< 0.001
	59.507	57.761	65.672	64.280	65.046	63.562	61.992	60.726	
4	61.594	1.228	67.231	0.999	66.673	1.006	63.584	1.055	< 0.001
	62.355	60.833	67.850	66.612	67.296	66.050	64.238	62.930	
5	63.337	1.019	69.073	1.293	68.687	1.191	65.494	0.887	< 0.001
	63.969	62.705	69.874	68.272	69.425	67.949	66.044	64.944	
6	65.229	1.302	70.640	1.618	69.746	1.688	66.981	1.582	< 0.001
	66.036	64.422	71.643	69.637	70.792	68.700	67.961	66.001	
7	66.426	1.148	71.585	1.049	70.872	1.074	67.839	0.620	< 0.001
	67.138	65.714	72.235	70.935	71.538	70.206	68.223	67.455	
8	67.523	1.415	72.762	1.610	72.167	1.476	68.898	1.445	< 0.001
	68.400	66.646	73.760	71.764	73.082	71.252	69.793	68.003	
9	68.566	1.174	73.559	1.088	72.795	1.103	69.909	1.150	< 0.001
	69.294	67.838	74.234	72.884	73.479	72.111	70.622	69.196	
10	69.686	1.269	74.311	1.073	73.576	1.205	70.453	1.252	< 0.001
	70.473	68.899	74.976	73.646	74.323	72.829	71.229	69.677	

Table 57: Means, standard deviations and 95% confidence levels of recall (word level)

Increment	Expert			Trained novice			Untrained novice		
	Mean difference	Bonferroni corrected *p*	Effect size (d)	Mean difference	Bonferroni corrected *p*	Effect size (d)	Mean difference	Bonferroni corrected *p*	Effect size (d)
1	-8.009	< 0.001	-5.981	-7.260	< 0.001	-5.132	-4.614	< 0.001	-3.327
2	-7.273	< 0.001	-4.974	-6.501	< 0.001	-4.255	-3.187	< 0.001	-2.028
3	-6.342	< 0.001	-5.245	-5.670	< 0.001	-4.572	-2.725	< 0.001	-2.334
4	-5.637	< 0.001	-5.308	-5.079	< 0.001	-4.769	-1.990	< 0.001	-1.832
5	-5.736	< 0.001	-5.194	-5.350	< 0.001	-5.088	-2.157	< 0.001	-2.380
6	-5.411	< 0.001	-3.884	-4.517	< 0.001	-3.159	-1.752	< 0.001	-1.275
7	-5.159	< 0.001	-4.945	-4.446	< 0.001	-4.216	-1.413	< 0.001	-1.614
8	-5.239	< 0.001	-3.644	-4.644	< 0.001	-3.386	-1.375	< 0.001	-1.013
9	-4.993	< 0.001	-4.650	-4.229	< 0.001	-3.914	-1.343	< 0.001	-1.218
10	-4.625	< 0.001	-4.149	-3.890	< 0.001	-3.314	-0.767	0.011	-0.641

Table 58: Mean difference, p-value and d-value of all increments of recall of systems compared to HGold (word level)

Bibliography

Aduriz, I., Aldezabal, I., Alegria, I., Arriola, J., Díaz de Ilarraza, A., Ezeiza, N. & Gojenola, K. 2003. Finite state applications for Basque. (*In* Proceedings of the 10th European Chapter of the Association for Computational Linguistics: Workshop on Finite-State Methods in Natural Language Processing, Budapest. Stroudsburg: Association for Computational Linguistics. p. 3-11).

Aha, D.W. 1992. Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies,* 36(2):267-287.

Ahern, S., Davis, M., Eckles, D., King, S., Naaman, M., Nair, R., Spasojevic, M. & Yang, J. 2006. Zonetag: designing context-aware mobile media capture to increase participation. (*In* Proceedings of the 8th International Conference on Ubiquitous Computing: Workshop on Pervasive Image Capture and Sharing, Orange County.).

Akasaka, R. 2009. Foreign accented speech transcription and accent recognition using a game-based approach. Swarthmore: Swarthmore College. (Thesis - BA).

Akkaya, C., Conrad, A., Wiebe, J. & Mihalcea, R. 2010. Amazon Mechanical Turk for subjectivity word sense disambiguation. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 195-203).

Alonso, O., Rose, D.E. & Stewart, B. 2008. Crowdsourcing for relevance evaluation. *Association for Computing Machinery Special Interest Group on Information Retrieval,* 42(2):9-15.

Anguera, X. & Oliver, N. 2008. MAMI: multimodal annotations on a camera phone. (*In* Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services, Lisbon. New York: Association for Computing Machinery. p. 379-382).

Audhkhasi, K., Georgiou, P. & Narayanan, S. 2011a. Reliability-weighted acoustic model adaptation using crowd sourced transcriptions. (*In* Cosi, P., De Mori, R., Di Fabbrizio, G. & Pieraccini, R. (*eds.*) Proceedings of the 12th Conference of the International Speech Communication Association, Florence. International Speech Communication Association. p. 3045-3048).

Audhkhasi, K., Georgiou, P. & Narayanan, S.S. 2011b. Accurate transcription of broadcast news speech using multiple noisy transcribers and unsupervised reliability metrics. (*In* Proceedings of the 36th Institute of Electrical and Electronics Engineers International Conference on Acoustics, Speech and Signal Processing, Prague. Red Hook: Institute of Electrical and Electronics Engineers. p. 4980-4983).

Bada, M., Eckert, M., Evans, D., Garcia, K., Shipley, K., Sitnikov, D., Baumgartner, W.A., Cohen, K.B., Verspoor, K. & Blake, J.A. 2012. Concept annotation in the CRAFT corpus. *BioMed Central Bioinformatics,* 13(1):161-181.

Barras, C., Geoffrois, E., Wu, Z.B. & Liberman, M. 2001. Transcriber: development and use of a tool for assisting speech corpora production. *Speech Communication,* 33(1):5-22.

Bergquist, M., Ljungberg, J. & Rolandsson, B. 2011. A historical account of the value of free and open source software: from software commune to commercial commons. (*In* Hissam, S.A., Russo, B., Neto, M.G. & Kon, F. (*eds.*) Proceedings of the International Federation for Information Processing: Advances in Information and Communication Technology Conference, Salvador. Heidelberg: Springer. p. 196-207).

Berntsson-Svensson, R. & Aurum, A. 2006. Successful software project and products: an empirical investigation. (*In* Proceedings of the Association for Computing Machinery and Institute for

Electrical and Electronics Engineers International Eymposium on Empirical Software Engineering, Rio de Janeiro. New York: Association for Computing Machinery. p. 144-153).

Bertran, M., Borrega, O., Recasens, M. & Soriano, B. 2008. AnCoraPipe: a tool for multilevel annotation. *Procesamiento del Lenguaje Natural,* 41(1):291-305.

Bilhaut, F. & Widlöcher, A. 2006. Linguastream: an integrated environment for computational linguistics experimentation. (*In* Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations, Trento. Stroudsburg: Association for Computational Linguistics. p. 95-98).

Bird, S. & Harrington, J. 2001. Speech annotation and corpus tools. *Speech Communication,* 33(1):1-4.

Boersma, P. 2002. Praat, a system for doing phonetics by computer. *Glot International,* 5(9):341-345.

Boves, L. & Oostdijk, N. 2003. Spontaneous speech in the spoken Dutch corpus. (*In* Proceedings of the Institute of Electrical and Electronics Engineers International Conference: Workshop on Spontaneous Speech Processing and Recognition, Tokyo.).

Callison-Burch, C. 2009. Fast, cheap, and creative: evaluating translation quality using Amazon's Mechanical Turk. (*In* Proceedings of the 14th Conference on Empirical Methods in Natural Language Processing, Singapore. Stroudsburg: Association for Computational Linguistics. p. 286-295).

Cardie, C. 2005. Machine learning for natural language processing (and vice versa?). (*In* Jorge, A.M., Torgo, L., Brazdil, P., Camacho, R. & Gama, J. (*eds.*) Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, Porto. Berlin: Springer-Verlag. p. 2-2).

Chamberlain, J., Poesio, M. & Kruschwitz, U. 2008. Phrase detectives: a web-based collaborative annotation game. (*In* Proceedings of the 5th International Conference on Semantic Systems, Graz.).

Chattopadhyay, R. 2013. Building adaptive computational systems for physiological and biomedical data via transfer and active learning. Tempe: Arizona State University. (Thesis - PhD).

Clark, A., Fox, C. & Lappin, S. (eds.) 2010. The handbook of computational linguistics and natural language processing. 1st ed. West Sussex: Wiley-Blackwell.

Cohen, J. 1988. Statistical power analysis for the behavioral sciences. 2nd ed. Hillsdale: Lawrence Erlbaum Associates.

Corston-Oliver, S.H. & Dolan, W.B. 1999. Less is more: eliminating index terms from subordinate clauses. (*In* Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, Maryland. Stroudsburg: Association for Computational Linguistics. p. 349-356).

Cunningham, H.M., Bontcheva, D. & Tablan, K. 2002. GATE: a framework and graphical development environment for robust NLP tools and applications. (*In* Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, Philadelphia. Stroudsburg: Association for Computational Linguistics. p. 168-175).

Daelemans, W., Zavrel, J., Van der Sloot, K. & Van den Bosch, A. 2001. TiMBL: Tilburg memory-based learner v4.0. Tilburg: Tilburg University.

Dandapat, S., Biswas, P., Choudhury, M. & Bali, K. 2009. Complex linguistic annotation - no easy way out!: a case from Bangla and Hindi POS labeling tasks. (*In* Proceedings of the 3rd Linguistic Annotation Workshop, Singapore. Stroudsburg: Association for Computational Linguistics. p. 10-18).

Dang, H.T., Chia, C.Y., Palmer, M. & Chiou, F.D. 2002. Simple features for Chinese word sense disambiguation. (*In* Proceedings of the 19th International Conference on Computational Linguistics, Taipei. Stroudsburg: Association for Computational Linguistics. p. 1-7).

Davel, M.H. & Barnard, E. 2005. Bootstrapping pronunciation dictionaries: practical issues. (*In* Proceedings of the 6th Conference of the International Speech Communication Association, Lisbon. Red Hook: International Speech Communication Association. p. 1561-1564).

Davel, M.H. & de Wet, F. 2010. Verifying pronunciation dictionaries using conflict analysis. (*In* Kobayashi, T., Hirose, K. & Nakamura, S. (*eds.*) Proceedings of the 11th Conference of the International Speech Communication Association, Makuhari. Red Hook: International Speech Communication Association. p. 1898-1901).

Davel, M.H., Van Heerden, C., Kleynhans, N. & Barnard, E. 2011. Efficient harvesting of internet audio for resource-scarce ASR. (*In* Cosi, P., De Mori, R., Di Fabbrizio, G. & Pieraccini, R. (*eds.*) Proceedings of the 12th Conference of the International Speech Communication Association, Florence. Red Hook: International Speech Communication Association. p. 27-31).

Day, D., McHenry, C., Kozierok, R. & Riek, L. 2004. Callisto: a configurable annotation workbench. (*In* Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon. Paris: European Language Resources Association. p. 2073-2076).

De Vries, N.J., Davel, M.H., Badenhorst, J., Basson, W.D., De Wet, F., Barnard, E. & De Waal, A. 2014. A smartphone-based ASR data collection tool for under-resourced languages. *Speech Communication,* 56:119-131.

De Wet, F. & De Vries, N. 2013. *Challenges of building speech corpora cheaply in the real world*. Paper presented at the Annual Conference of the Pattern Recognition Association of South Africa: Resource Management Agency Workshop. [Unpublished].

De Wet, F., Louw, P. & Niesler, T. 2006. The design, collection and annotation of speech databases in South Africa. (*In* Proceedings of the 17th International Symposium of the Pattern Recognition Association of South Africa, Parys. Pretoria: Pattern Recognition Association of South Africa. p. 61-65).

Denis, P. & Sagot, B. 2009. Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art POS tagging with less human effort. (*In* Kwong, O. (*ed.*) Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation Conference, Hong Kong. Hong Kong: City University of Hong Kong Press. p. 110-119).

Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., Tomkins, A. & Tomlin, J.A. 2003. SemTag and Seeker: bootstrapping the semantic web via automated semantic annotation. (*In* Proceedings of the 12th International Conference on World Wide Web, Budapest. New York: Association for Computing Machinery. p. 178-186).

Dligach, D., Nielsen, R.D. & Palmer, M. 2010. To annotate more accurately or to annotate more. (*In* Proceedings of the 4th Linguistic Annotation Workshop, Upsala. Stroudsburg: Association for Computational Linguistics. p. 64-72).

Eickhoff, C. & de Vries, A. 2011. How crowdsourcable is your task. (*In* Proceedings of the 4th Association for Computing Machinery International Conference on Web Search and Data Mining: Workshop on Crowdsourcing for Search and Data Mining, Hong Kong. New York: Association for Computing Machinery. p. 11-14).

Ellis, S.M. & Steyn, H.S. 2003. Practical significance (effect sizes) versus or in combination with statistical significance (p-values). *Management Dynamics,* 12(4):51-53.

Eryigit, G. 2007. ITU Treebank Annotation Tool. (*In* Proceedings of the 1st Linguistic Annotation Workshop, Prague. Stroudsburg: Association of Cognitive Linguistics. p. 117-120).

Evanini, K., Higgins, D. & Zechner, K. 2010. Using Amazon Mechanical Turk for transcription of non-native speech. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and

Language Data with Amazon's Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 53-56).

Finin, T., Murnane, W., Karandikar, A., Keller, N., Martineau, J. & Dredze, M. 2010. Annotating named entities in twitter data with crowdsourcing. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 80-88).

Fuggetta, A. 2003. Open source software - an evaluation. *Journal of Systems and Software,* 66(1):77-90.

Gao, Q. & Vogel, S. 2010. Consensus versus expertise: a case study of word alignment with mechanical turk. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 30-34).

Geertzen, J., Petukhova, V. & Bunt, H. 2008. Evaluating dialogue act tagging with naive and expert annotators. (*In* Proceedings of the 5th International Conference on Language Resources and Evaluation, Marrakech. p. 1076-1082).

Gelas, H., Abate, S.T., Besacier, L. & Pellegrino, F. 2011. Quality assessment of crowdsourcing transcriptions for African languages. (*In* Proceedings of the 12th Conference of the International Speech Communication Association. Red Hook: International Speech Communication Association. p. 3065-3068).

Germann, U. 2007. Two tools for creating and visualizing sub-sentential alignments of parallel text. (*In* Proceedings of the 1st Linguistic Annotation Workshop, Prague. Stroudsburg: Association for Computational Linguistics. p. 121-124).

Green, D.C. 2011. Developing an energy information system: custom design vs. off-the-shelf software. (*In* Capehart, B.L. & Middelkoop, T. (*eds.*). Handbook of Web Based Energy Information and Control Systems. Lilburn: Fairmont Press. 349-352).

Groenewald, H.J. 2006. Automatic lemmatisation for Afrikaans. Potchefstroom: North-West University. (Dissertation - MEng).

Grover, A.S., Van Huyssteen, G.B. & Pretorius, M.W. 2010. An HLT profile of the official South African languages. (*In* Proceedings of the 2nd Workshop on African Language Technology, Valletta. Paris: European Language Resources Association. p. 3-7).

Hall, M.A. & Smith, L.A. 1998. Practical feature subset selection for machine learning. (*In* Proceedings of the 21st Australasian Computer Science Conference, Perth. p. 181-191).

Hand, D.J. 2007. Principles of data mining. *Drug Safety,* 30(7):621-622.

Hardy, H., Baker, K., Bonneau-Maynard, H., Devillers, L., Rosset, S. & Strzalkowski, T. 2003. Semantic and dialogic annotation for automated multilingual customer service. (*In* Proceedings of the 8th Conference of the International Speech Communication Association, Geneva. p. 201-204).

Heilman, M. & Smith, N.A. 2010. Rating computer-generated questions with Mechanical Turk. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Los Angeles. Stroudsburg: Association for Computional Linguistics. p. 35-40).

Higgins, C., McGrath, E. & Moretto, L. 2010. MTurk crowdsourcing: a viable method for rapid discovery of Arabic nicknames? (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 89-92).

Hong, J. & Baker, C.F. 2011. How good is the crowd at “real” WSD? (*In* Proceedings of the 5th Linguistic Annotation Workshop, Portland. Stroudsburg: Association for Computational Linguistics. p. 30-37).

Howe, J. 2008. Crowdsourcing: why the power of the crowd is driving the future of business. 1st ed. New York: Crown Publishing Group.

Hsueh, P.Y., Melville, P. & Sindhwani, V. 2009. Data quality from crowdsourcing: a study of annotation selection criteria. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technology: Workshop on Active Learning for Natural Language Processing, Boulder. Stroudsburg: Association for Computational Linguistics. p. 27-35).

Ipeirotis, P. 2010. Demographics of mechanical turk. *Center for Digital Economy Research Working Paper Series,* 10(1):1-14.

Irvine, A. & Klementiev, A. 2010. Using Mechanical Turk to annotate lexicons for less commonly used languages. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 108-113).

Jha, M., Andreas, J., Thadani, K., Rosenthal, S. & McKeown, K. 2010. Corpus creation for new genres: a crowdsourced approach to PP attachment. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 13-20).

Kalapanidas, E., Avouris, N., Craciun, M. & Neagu, D. 2003. Machine learning algorithms: a study on noise sensitivity. (*In* Proceedings of the 1st Balkan Conference in Informatics, Thessaloniki. p. 356-365).

Khoshgoftaar, T.M., Van Hulse, J. & Napolitano, A. 2010. Supervised neural network modeling: an empirical investigation into learning from imbalanced data with labeling errors. *Institute of Electrical and Electronics Engineers: Transactions on Neural Networks,* 21(5):813-830.

Kipp, M. 2008. Spatiotemporal coding in ANVIL. (*In* Proceedings of the 6th International Language Resources and Evaluation Conference, Marrakech. European Language Resources Association. p. 2042-2045).

Kogut, P. & Holmes, W. 2001. AeroDAML: Applying information extraction to generate DAML annotations from web pages. (*In* Proceedings of the 1st International Conference on Knowledge Capture: Workshop on Knowledge Markup and Semantic Annotation, Victoria. New York: Association for Computing Machinery. p. 2-2).

Krauwer, S. 2003. The basic language resource kit (BLARK) as the first milestone for the language resources roadmap. (*In* Proceedings of 15th International Conference on Speech and Computer, Pilsen. Chennai: Springer. p. 8-15).

Lawson, N., Eustice, K., Perkowitz, M. & Yetisgen-Yildiz, M. 2010. Annotating large email datasets for named entity recognition with Mechanical Turk. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 71-79).

Lee, C. & Glass, J. 2011. A transcription task for crowdsourcing with automatic quality control. (*In* Cosi, P., De Mori, R., Di Fabbrizio, G. & Pieraccini, R. (*eds.*) Proceedings of the 12th Conference of the International Speech Communication Association, Florence. Red Hook: International Speech Communication Association. p. 3041-3044).

Lewis, M.P. 2009. Ethnologue: languages of the world. 16th ed. [Online]. Available: http://www.ethnologue.com Date of access: 15 Feb. 2013.

Li, J.Y., Conradi, R., Slyngstad, O.P.N., Bunse, C., Torchiano, M. & Morisio, M. 2009. Development with off-the-shelf components: 10 facts. *Institute of Electrical and Electronics Engineers Software,* 26(2):80-87.

Maamouri, M. & Bies, A. 2004. Developing an Arabic treebank: methods, guidelines, procedures, and tools. (*In* Proceedings of the 20th International Conference on Computational Linguistics: Workshop on Computational Approaches to Arabic Script-based Languages, Geneva. Stroudsburg: Association for Computational Linguistics. p. 2-9).

Maeda, K., Lee, H., Medero, J. & Strassel, S. 2006. A new phase in annotation tool development at the linguistic data consortium: the evolution of the annotation graph toolkit. (*In* Proceedings of the 5th International Conference on Language Resources and Evaluation, Genoa. p. 1570-1573).

Marge, M., Banerjee, S. & Rudnicky, A.I. 2010a. Using the Amazon Mechanical Turk for transcription of spoken language. (*In* Proceedings of the 35th Institute of Electrical and Electronics Engineers International Conference on Acoustics, Speech, and Signal Processing, Dallas. Red Hook: Institute of Electrical and Electronics Engineers p. 5270-5273).

Marge, M., Banerjee, S. & Rudnicky, A.I. 2010b. Using the Amazon Mechanical Turk to transcribe and annotate meeting speech for extractive summarization. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 99-107).

McGraw, I., Lee, C., Hetherington, L., Seneff, S. & Glass, J. 2010. Collecting voices from the cloud. (*In* Proceedings of the 7th Language Resources and Evaluation Conference, Valletta. p. 19-21).

McKelvie, D., Isard, A., Mengel, A., Moller, M.B., Grosse, M. & Klein, M. 2001. The MATE workbench - an annotation tool for XML coded speech corpora. *Speech Communication,* 33(1):97-112.

McKinney, D. 2001. Impact of commercial-off-the-shelf (COTS) software and technology on systems engineering. *Presentation to the International Council on Systems Engineering Chapters.* p. 1-19.

Mellebeek, B., Benavent, F., Grivolla, J., Codina, J., Costa-Jussa, M.R. & Banchs, R. 2010. Opinion mining of spanish customer comments with non-expert annotations on Mechanical Turk. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 114-121).

Milde, J.T. & Gut, U. 2002. The TASX-environment: an XML-based toolset for time aligned speech corpora. (*In* Proceedings of the 3rd Language Resources and Evaluation Conference, Las Palmas. p. 1922-1927).

Min, B. 2013. Relation extraction with weak supervision and distributional semantics. New York: New York University. (Thesis - PhD).

Modipa, T.I., Davel, M.H. & De Wet, F. 2013. Implications of Sepedi/English code switching for ASR systems. (*In* Proceedings of the 24th Annual Symposium of the Patern Recognition Association of South Africa, Johannesburg. p. 64-69).

Müller, C. & Strube, M. 2001. MMAX: a tool for the annotation of multi-modal corpora. (*In* Proceedings of the 2nd International Joint Conferences on Artificial Intelligence: Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Seattle. San Francisco: Morgan Kaufmann Publishers. p. 45-50).

Munro, R., Bethard, S., Kuperman, V., Lai, V.T., Melnick, R., Potts, C., Schnoebelen, T. & Tily, H. 2010. Crowdsourcing and language studies: the new generation of linguistic data. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 122-130).

Munro, R. & Tily, H. 2011. The start of the art: an introduction to crowdsourcing technologies for language and cognition studies. (*In* Proceedings of the Workshop on Crowdsourcing Technologies for Language and Cognition Studies, Boulder. p. 1-10).

Nettleton, D.F., Orriols-Puig, A. & Fornells, A. 2010. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review,* 33(4):275-306.

Novotney, S. & Callison-Burch, C. 2010. Cheap, fast and good enough: automatic speech recognition with non-expert transcription. (*In* Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 207-215).

O'Donnell, M. 2000. RSTTool 2.4: a markup tool for rhetorical structure theory. (*In* Proceedings of the 1st International Conference on Natural Language Generation, Mitzpe Ramon. Stroudsburg: Association for Computational Linguistics. p. 253-256).

Odendal, F.F. & Gouws, R.H. 2005. HAT: handwoordeboek van die Afrikaanse taal. 5th ed. Pinelands: Maskew Miller Longman.

Oosthuizen, N.L., Puttkammer, M.J. & Schlemmer, M. 2010. Improving orthographic transcriptions of speech corpora. (*In* De Pauw, G., Groenewald, H.J. & De Schryver, G. (*eds.*) Proceedings of the 7th International Conference on Language Resources and Evaluation: 2nd Workshop on African Language Technology, Valletta. European Language Resources Association. p. 55-58).

Orasan, C. 2003. PALinkA: a highly customisable tool for discourse annotation. (*In* Proceedings of the 4th Special Interest Group on Discourse and Dialogue: Workshop on Discourse and Dialogue, Sapporo. p. 39-43).

Palmer, M., Chiou, F.D., Xue, N. & Lee, T.K. 2005a. Chinese Treebank 5.0. *Catalog number: LDC2005T01* [Online]. Philadelphia: Linguistic Data Consortium. Available: http://catalog.ldc.upenn.edu/LDC2005T01 Date of access: 17 May 2012.

Palmer, M., Kingsbury, P. & Gildeafi, D. 2005b. The proposition bank: an annotated corpus of semantic roles. *Computational Linguistics,* 31(1):71-106.

Parent, G. & Eskenazi, M. 2010. Toward better crowdsourced transcription: transcription of a year of the let's go bus information system data. (*In* Proceedings of the Institute of Electrical and Electronics Engineers Conference: Spoken Language Technology Workshop, Berkeley. Red Hook: Institute of Electrical and Electronics Engineers. p. 312-317).

Plaehn, O. & Brants, T. 2000. Annotate - an efficient interactive annotation tool. (*In* Proceedings of the 6th Conference on Applied Natural Language Processing, Seattle. Stroudsburg: Association for Computational Linguistics).

Rebbapragada, U. & Brodley, C. 2007. Class noise mitigation through instance weighting. (*In* Proceedings of the 18th European Conference on Machine Learning, Warsaw. Berlin: Springer. p. 708-715).

Rose, T., Stevenson, M. & Whitehead, M. 2002. The Reuters corpus volume 1 - from yesterday's news to tomorrow's language resources. (*In* Proceedings of the 3rd Language Resources and Evaluation Conference, Las Palmas. p. 827-832).

Ross, J., Irani, L., Silberman, M., Zaldivar, A. & Tomlinson, B. 2010. Who are the crowdworkers?: shifting demographics in mechanical turk. (*In* Proceedings of the 28th Conference on Human Factors in Computing Systems: Extended Abstracts on Human Factors in Computing Systems, Atlanta. New York: Association for Computing Machinery. p. 2863-2872).

Roux, J.C., Louw, P.H. & Niesler, T. 2004. The African speech technology project: an assessment. (*In* Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon. Paris: European Language Resources Association. p. 93-96).

Roy, B.C., Vosoughi, S. & Roy, D. 2010. Automatic estimation of transcription accuracy and difficulty. (*In* Kobayashi, T., Hirose, K. & Nakamura, S. (*eds.*) Proceedings of the 11th Conference of the International Speech Communication Association, Makuhari. Red Hook: International Speech Communication Association. p. 1902-1905).

Rumshisky, A. 2011. Crowdsourcing word sense definition. (*In* Proceedings of the 5th Linguistic Annotation Workshop, Portland. Stroudsburg: Association for Computational Linguistics. p. 74-81).

Russell, B.C., Torralba, A., Murphy, K.P. & Freeman, W.T. 2008. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision,* 77(1):157-173.

Schmidt, T. & Wörner, K. 2008. EXMARaLDA - creating, analysing and sharing spoken language corpora for pragmatic research. *Pragmatics,* 19(4):565-582.

Schmitt, N. & McCarthy, M. 1997. Vocabulary: Description, acquisition and pedagogy. Cambridge: Cambridge university press.

Sheng, V.S., Provost, F. & Ipeirotis, P.G. 2008. Get another label?: improving data quality and data mining using multiple, noisy labelers. (*In* Proceedings of the 14th Association for Computing Machinery Special Interest Group on Knowledge Discovery and Data Mining: International Conference on Knowledge Discovery and Data Mining Las Vegas. New York: Association for Computing Machinery. p. 614-622).

Silberztein, M. 2005. NooJ: a linguistic annotation system for corpus processing. (*In* Proceedings of the Human Language Technologies and Empirical Methods in Natural Language Processing Conference: Demonstration Abstracts, Vancouver. Stroudsburg: Association for Computational Linguistics. p. 10-11).

Sjölander, K. & Beskow, J. 2000. Wavesurfer - an open source speech tool. (*In* Proceedings of the 6th International Conference on Spoken Language Processing, Beijing. p. 464-467).

Skory, A. & Eskenazi, M. 2010. Predicting cloze task quality for vocabulary training. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Innovative Use of Natural Language Processing for Building Educational Applications Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 49-56).

Snow, R., O'Connor, B., Jurafsky, D. & Ng, A.Y. 2008. Cheap and fast - but is it good?: evaluating non-expert annotations for natural language tasks. (*In* Proceedings of the 13th Conference on Empirical Methods in Natural Language Processing, Waikiki. Stroudsburg: Association for Computational Linguistics. p. 254-263).

Stamelos, I., Angelis, L., Morisio, M., Sakellaris, E. & Bleris, G.L. 2003. Estimating the development cost of custom software. *Information & Management,* 40(8):729-741.

Statistics South Africa. 2013. Census 2011: census in brief. Pretoria: Statistics South Africa.

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S. & Tsujii, J.i. 2012. BRAT: a web-based tool for NLP-assisted text annotation. (*In* Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics, Avignon. Stroudsburg: Association for Computational Linguistics. p. 102-107).

Strassel, S., Kolár, J., Song, Z., Barclay, L. & Glenn, M. 2005. Structural metadata annotation: moving beyond English. (*In* Proceedings of the 6th Conference of the International Speech Communication Association, Lisbon. Red Hook: International Speech Communication Association. p. 1545-1548).

Stuckless, R. 1994. Developments in real-time speech-to-text communication for people with impaired hearing. (*In* Ross, M. (*ed.*). Communication Access for People with Hearing Loss: Compliance with the Americans with Disabilities Act. Baltimore: York Press. 198-226).

Tomanek, K., Wermter, J. & Hahn, U. 2007. Efficient Annotation with the Jena ANnotation Environment (JANE). (*In* Proceedings of the 1st Linguistic Annotation Workshop, Prague. Stroudsburg: Association for Computational Linguistics. p. 9-16).

Van Huyssteen, G.B. & Puttkammer, M.J. 2007. Accelerating the annotation of lexical data for less-resourced languages. (*In* Proceedings of the 8th Conference of the International Speech Communication Association, Antwerp. Red Hook: International Speech Communication Association. p. 1505-1508).

Verhagen, M. 2010. The Brandeis annotation tool. (*In* Proceedings of the 7th International Conference on Language Resources and Evaluation, Malta. p. 3638-3643).

Vigder, M., Gentleman, W.M. & Dean, J. 2010. COTS software integration: state of the art. National Research Council of Canada, Institute for Information Technology.

Voas, J. 1998. COTS software: the economical choice? *Institute of Electrical and Electronics Engineers Software,* 15(2):16-19.

Wagacha, P.W., De Pauw, G. & Getao, W. 2006. Development of a corpus for Gĩkũyũ using machine learning techniques. (*In* Roux, J.C. (*ed.*) Proceedings of the 5th Language Resources and Evaluation Conference: Workshop on Networking the Development of Language Resources for African Languages, Genoa. European Language Resources Association. p. 170-179).

Wald, M. 2011. Crowdsourcing correction of speech recognition captioning errors. (*In* Proceedings of the 8th International Cross-disciplinary Conference on Web Accessibility, Hyderabad. New York: Association for Computing Machinery. p. 22-23).

Wang, Y. & Li, Q. 2013. Review on the studies and advances of machine learning approaches. *TELKOMNIKA Indonesian Journal of Electrical Engineering,* 12(2).

Whittaker, E.W.D. & Woodland, P.C. 2001. Efficient class-based language modelling for very large vocabularies. (*In* Proceedings of the Institute of Electrical and Electronics Engineers International Conference on Acoustics, Speech, and Signal Processing. Red Hook: Institute of Electrical and Electronics Engineers. p. 545-548).

Xue, N. & Zhou, Y. 2010. Applying syntactic, semantic and discourse constraints in Chinese temporal annotation. (*In* Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Beijing. Stroudsburg: Association for Computational Linguistics. p. 1363-1372).

Yetisgen-Yildiz, M., Solti, I., Xia, F. & Halgrim, S.R. 2010. Preliminary experience with Amazon's Mechanical Turk for annotating medical named entities. (*In* Proceedings of the 11th North American Chapter of the Association for Computational Linguistics Human Language Technologies: Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk, Los Angeles. Stroudsburg: Association for Computational Linguistics. p. 180-183).

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D. & Valtchev, V. 2009. The HTK book (for HTK version 3.4). Cambridge: Cambridge University Engineering Department.

Zaghouani, W., Diab, M., Mansouri, A., Pradhan, S. & Palmer, M. 2010. The revised Arabic propbank. (*In* Proceedings of the 4th Linguistic Annotation Workshop, Uppsala. Stroudsburg: Association for Computational Linguistics. p. 222-226).

Zaidan, O.F. & Callison-Burch, C. 2011. Crowdsourcing translation: professional quality from non-professionals. (*In* Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics Human Language Technologies, Portland. Stroudsburg: Association for Computational Linguistics. p. 1220-1229).

Zhu, X., Wu, X. & Chen, Q. 2003. Eliminating class noise in large datasets. (*In* Fawcett, T. & Mishra, N. (*eds.*) Proceedings of the 20th International Conference on Machine Learning, Washington DC. Menlo Park: The Association for the Advancement of Artificial Intelligence Press. p. 920-927).

Zhu, X.Q. & Wu, X.D. 2004. Class noise vs. attribute noise: a quantitative study of their impacts. *Artificial Intelligence Review,* 22(3):177-210.