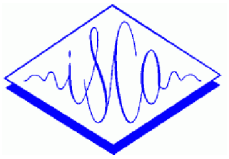


Third Workshop on Spoken Language
Technologies for Under-resourced Languages
Cape Town, South Africa
May 7-9, 2012

WEB-BASED CORPUS ACQUISITION FOR SWAHILI LANGUAGE MODELLING

Alexander Kivaisi and Audrey Mbogho
Department of Computer Science
University of Cape Town
Alexander.Kivaisi@uct.ac.za Audrey.Mbogho@uct.ac.za

ABSTRACT

Finding large amounts of text data for use in natural language technology is difficult for under-resourced languages such as Swahili. The corpora that are readily accessible for these languages are not sufficient to be used in language technologies, whose requirements can run into the hundreds of millions of words. This paper describes how we can take advantage of search engines such as Google together with crawling tools to collect Swahili text from the Web. We also share the experience of cleaning up and normalising the resulting text data. Finally, we show some preliminary results of the evaluation of the language models built from our corpus as well as results of how they compare to those built from the Helsinki Corpus.

Index Terms— Under-resourced languages, corpus acquisition, Swahili, language model

1. INTRODUCTION

Natural language technology applications, such as, speech recognition, machine translation, and handwriting recognition, require large amounts of training text data. For such applications performance normally increases with increasing amount of training data. In large vocabulary continuous speech recognition systems [4], hundreds of millions of words are needed to obtain accurate probability estimation for the statistical language model. In [1], Banko and Brill showed that for context sensitive spelling correction, increasing the training data size increases the accuracy for up to 1 billion words. Likewise in machine translation, as described in [2], the more the training data, the better the results.

Although the Web is growing tremendously all the time, with large amounts of text data being added to it daily, finding contents for a particular target language can be difficult. First, it requires knowing the location of these contents and how to collect them. As there are millions and millions of Web pages, knowing which web site has content in a given language is an enormous challenge. This situation is clearly most severe for under-resourced languages. Secondly, the process of cleaning up and normalizing the contents is always crucial and needs extra care so as to achieve high performance in the resulting language model. This involves text-processing tasks such as removing markup tags or unwanted text, clearing empty spaces or lines to produce one-line single sentences, and expanding abbreviations and acronyms. In particular, normalisation [4] such as converting non-words into spoken words depends on the target language, requiring some understanding of the language so that the conversion rules can be defined and word ambiguity can be resolved. Therefore, existing scripts which have automated these tasks for other languages may not be helpful.

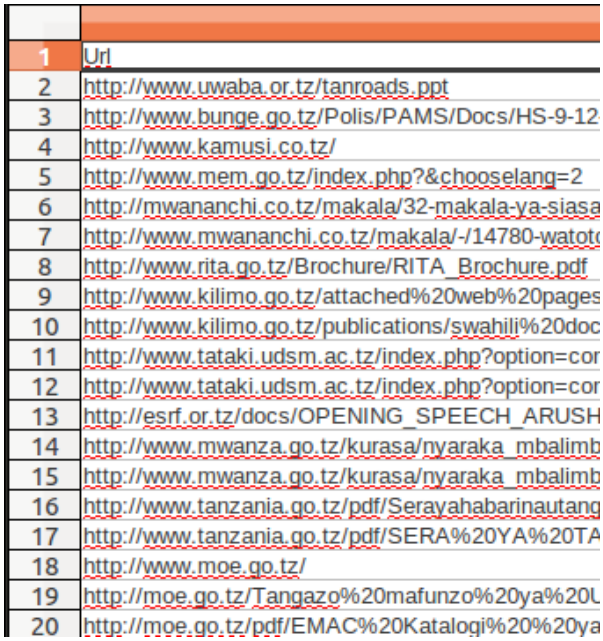
The focus of this study is Swahili, an under-resourced language spoken by over 100 million people in East and Central Africa, mainly Tanzania, Kenya, the Comoros, Rwanda, Burundi, Uganda and the DRC, and to a lesser extent, Malawi, Zambia, Mozambique and Somalia (see http://swahililanguage.stanford.edu/). For this reason, one can approach the Web with a certain level of confidence about the presence of significant Swahili content there. The next section shows how the search engine Google was used to locate Swahili contents. Section 3 shows how these contents were collected by crawling into the web sites found by the search. Cleaning up of the contents is explained in Section 4, while Section 5 shares the experience of developing the normalization rules. Section 6 shows some preliminary results on the language model, while Section 7 discusses these results with reference to previous work.

2. WEB SEARCH

Searching for content on the Internet has always been difficult because of the distributed manner in which the content is stored. Because English dominates the Web, content in less popular languages is hard to discover.

The search engine Google contains features that help to narrow down the search results. The language selection feature made it possible to list documents or domain names written in Swahili. For our search queries, we used the more common Swahili keywords ("na", "ya", "ni", meaning “and”, “of”, “is”) which are likely to be found in any Swahili document. SEOquake, a FireFoxaddon, helped to export the list of URLs in the search results into csv files (see Figure 1). These csv files were then merged into a text file and URLs were truncated remaining with only root domain names. Redundant domain names were then removed from the list. The purpose of retaining only domain names was to enable the web crawler to visit the

entire website by starting from the root and passing through all the pages hosted on that domain. The whole process of merging and truncating was done using a Perl script.



	Url
1	
2	http://www.uwaba.or.tz/tanroads.ppt
3	http://www.bunge.go.tz/Polis/PAMS/Docs/HS-9-12
4	http://www.kamusi.co.tz/
5	http://www.mem.go.tz/index.php?&chooselang=2
6	http://mwananchi.co.tz/makala/32-makala-ya-siasa
7	http://www.mwananchi.co.tz/makala/-/14780-watot
8	http://www.rita.go.tz/Brochure/RITA_Brochure.pdf
9	http://www.kilimo.go.tz/attached%20web%20pages
10	http://www.kilimo.go.tz/publications/swahili%20doc
11	http://www.tataki.udsm.ac.tz/index.php?option=cor
12	http://www.tataki.udsm.ac.tz/index.php?option=cor
13	http://esrf.or.tz/docs/OPENING_SPEECH_ARUSH
14	http://www.mwanza.go.tz/kurasa/nyaraka_mbalimb
15	http://www.mwanza.go.tz/kurasa/nyaraka_mbalimb
16	http://www.tanzania.go.tz/pdf/Serayahabarinautang
17	http://www.tanzania.go.tz/pdf/SERA%20YA%20TA
18	http://www.moe.go.tz/
19	http://moe.go.tz/Tangazo%20mafunzo%20ya%20U
20	http://moe.go.tz/pdf/EMAC%20Katalogi%20%20ya

Figure 1. Search results exported usingSEOQuake

We performed a manual cross-check, looking up one URL after another to see how much Swahili content exists on those web sites. It may be argued that we could have narrowed down our search results by just focusing on domain names of Swahili-speaking countries, for instance domains ending in .tz, meaning URLs from Tanzania where one is most likely to find content written only in Swahili. Unfortunately, this was not the case since it turned out that some of these web sites contain content in both English and Swahili and, furthermore, the percentage of content that is in English is sometimes much greater than the Swahili content. Thus, domain names by themselves are not a reliable guide on the language of the content. A trigram language model can alleviate the task of detecting documents of the desired type; however, there was none available to us, and, in fact, the ultimate goal of this work is to build a Swahili language model. A total of 4220 URLs were found which had any number of Swahili words. Some URLS had the same root domain name and in such cases, only one such URL was retained. The number after discarding redundant domain names was 1498. Lastly, a manual check was conducted for the amount of Swahili content. Only a final 36 domains were found to have a significant number of Swahili sentences (proper or improper).This final list was then used in the crawling tool to download all the content.

3. COLLECTING THE CONTENT

For collecting the text data we used **wget** since it provides all the necessary features required to perform the work efficiently (see below for how the command was used to launch this process). The command **wget** is a free tool for retrieving content from the web. It supports downloading via HTTP, HTTPS and FTP protocols. Its robustness allows it to work over slow and unstable network connections. It can work as a Web crawler by downloading recursively the resources linked to each other. It also allows selection of document types to download which is crucial when there is limited bandwidth and storage.

wget -r -Nc -np -R gif,jpg,jpeg,js,png,JPG,FLV,MP4,AVI,css,ico,flv,swf,mp4, avi,mp3,mov -i ../list.txt

- ***-r :perform recursive download***
- ***-Nc :skipping downloads that would download to the existing files***
- ***-np : don't ascend to the parent directory***
- ***-R : Reject files extension***
- ***-I : download URLs found in a file***

The final 36 domain names list file was then split into five text files for simplicity, each containing at least 6 domain names. These files were then used one by one in the **wget** command to retrieve the content from each of the domain names. Since we were only interested in textual information, pictures, videos, audio and other arbitrary binary content were avoided. The process of downloading the content from the domains on each text file took about a day. The total download size of all content was approximately 1.9GB of textual content, where most consisted of html files and some consisted of pdf files, word documents and text files.

4. CLEANING

Before starting with cleaning up the contents, all web pages retrieved from a particular domain were merged into a single text file. This made the cleaning process much quicker and easier than if it were to be done for hundreds of html files individually. By merging this way, each domain name corresponded to a single text file for clean-up.

Since the conversion of pdf and word documents to text files requires other tools and requires more text processing to come up with clearly defined Swahili sentences, we decided to extract the contents from html files which seemed easier but also sufficient. Instead of removing the web based tags we decided to extract the contents from certain tags, which made it possible to retrieve more useful content. Tags like <p>,
, ,<th>were used to

extract groups of linguistic sentences, while tags like <a>, <script>,<style> were ignored as they contain presentation information, and not content. Non-mapped Unicode characters, empty lines and empty spaces were all removed. Punctuation marks such as (.?!) were used to identify sentences which were then listed line by line in a text file. Other characters such as (*()[]{}"#><) were also removed

Furthermore, a Perl script was used to identify and discard sentences with more than fifty percent of words which appeared not to be in the Swahili dictionary. The Swahili dictionary was created from the Helsinki Corpus which contains more Swahili words than the Freedict, Tuki or TeDje-SED dictionaries [3]. The presence of such sentences was because at times splitting of sentence at markers was not done perfectly which left some grammatically incorrect Swahili sentences and some that were merged with English words. Again not all the contents were just in Swahili; some were a mixture of both languages which made it difficult to filter out. However, our primary goal was to collect as many Swahili sentences as possible regardless of ending up possibly with 50% of English words in the sentence. The minimum sentence length was about 4 words while the maximum length was about 99 words. Table 1shows the data counts during the cleaning stage.

Table 1. Statistics for cleaned-up web corpus

Total number of sentences	488,273
Max number of words per sentence	99
Min number of words per sentence	4

5. NORMALISATION

Normalization is the process of converting non-standard words, which includes numbers, currency, abbreviations, acronyms and dates, into spoken words. This process is very crucial when it comes to applying the language model in applications such as automatic speech recognition in order to have better recognition results. Leaving these non-standard words will introduce noise in the training data causing the language model not to perform well when applied to speech recognition. Although the set of non-standard words varies from one language to another, fortunately, they can be handled in a manner that is independent of language. The process, mostly involves a standard set of steps, which can be applied to any language. However, resolution of word ambiguity can demand more attentionas it depends on the word context, and requires familiarity with the language itself, which can be hard sometimes depending on the dimensions of the problem.

The process of normalization begins with splitting the input text into tokens, followed by identifying the types of non-standard words and their categories. It then finishes with performing the expansion. In this work, we started with the straightforward identification of these types and performing the expansion. Tokenization was performed separately during the cleaning up. Perl regular expressions were used to define the rules of identifying the categories. We adopted similar rules to those used in [9].The order of identifying the types of category was important in order to deal with ambiguity. For instance, dates with this format (12/23/2002) had to be detected first before fractions (12/23). The following non-standard word types were chosen for normalization: time, numbers (cardinal, fractions, and decimals), emails, web addresses, telephone numbers and fax numbers, abbreviations and acronyms (see Table 2 for some examples).

Table 2.Normalisation examples

Category	Format	Conversion (Swahili example)	Conversion (English example)
Email	Abc_123 @ yahoo.com	A b c underscore mojambilitatu at yahoo nukta com	A b c underscore one two three at yahoo dot com
Times	**saa 12:30**	Saasitananusu	Half past twelve
Dates	**30-02-2002** or	Thelathinimwezi wapilimwaka elfumbilinambili	Thirtieth of February two thousand and two
Decimals	**3.4**	Tatunuktanne	Three point four
Fractions	**½**	nusu	half
Cardinal	**34**	Thelathininanne	Thirty four

6. LANGUAGE MODELS

We built different types of n-gram language model for both the text prepared from the web and Helsinki corpus [7]. The text from the Helsinki corpus, to our knowledge the only large tag Swahili corpus currently available online, was prepared in a similar fashion to that from the web. Sentence separation and cleaning up of empty lines or spaces was already done, which made the task of cleaning up much easier.

We used the SRILM toolkit [8] to build the languages models. These language models were built using different types of smoothing techniques and different sizes of the Swahili dictionary (20K, 50K, 64K and 95K) in order to analyse the effect of both on the language model by computing the model's perplexity. Ultimately the goal of

this work is to evaluate the language models according to their performance in machine translation tasks, but this is left to future work.

We also combined the two sets of data in order to analyse the performance of the language model as we increased the size of the text data set. Figures 2 to 11 show the experimental results, first on comparison between different smoothing techniques and finally on performance of each language model as the size of the data set increases.

The following smoothing techniques were applied:

- GT Good Turing
- AB Absolute Discounting
- WB Witten-Bell Discounting
- RD Ristard's Natural Discounting
- O – KN Original Kneser-Ney Discounting
- M-KN Modified Kneser-Ney Discounting
- O-KN-I Original Kneser-Ney interpolated

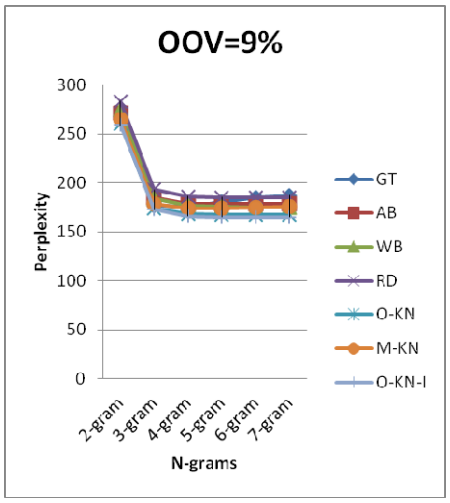


Figure 2. Helsinki corpus, 20K dictionary

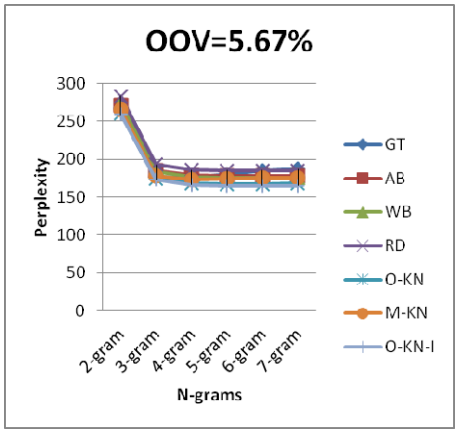


Figure 3. Helsinki corpus, 50K dictionary

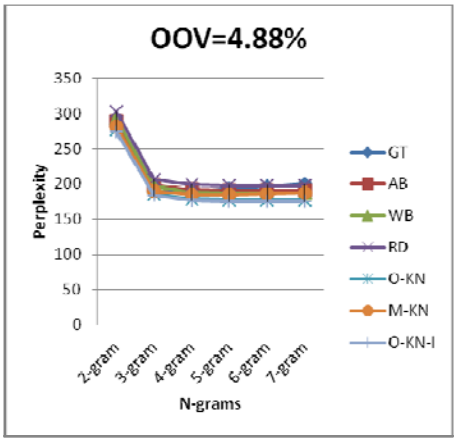


Figure 4. Helsinki corpus, 64K dictionary

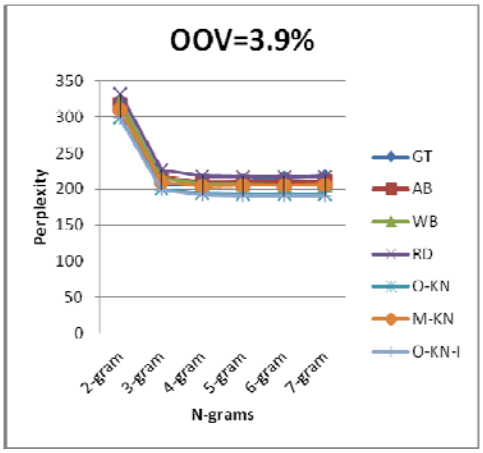


Figure 5. Helsinki corpus, 95K dictionary

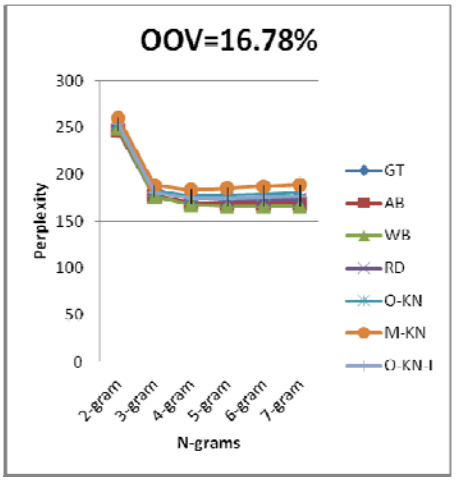


Figure 6. Web corpus, 20K dictionary

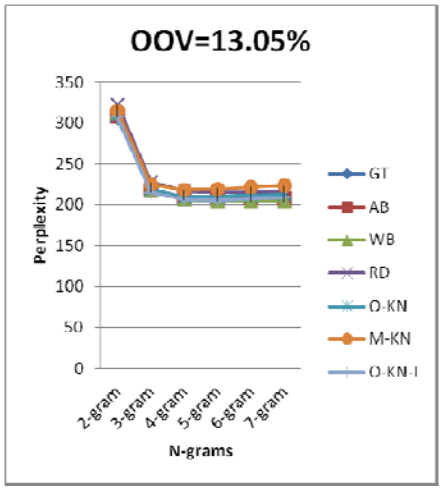


Figure 7. Web corpus, 50K dictionary

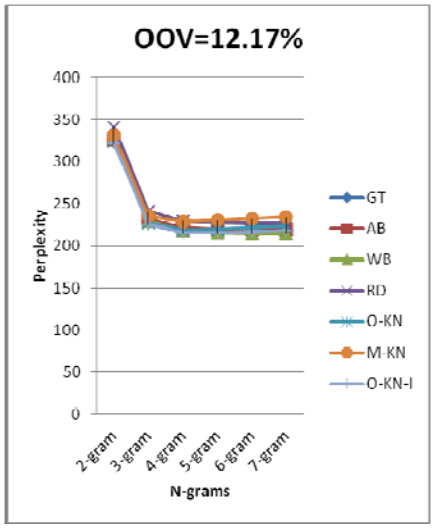


Figure 8. Web corpus, 64K dictionary

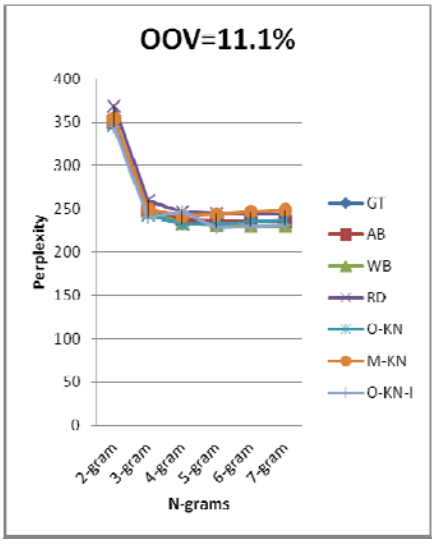


Figure 9. Web corpus, 95K dictionary

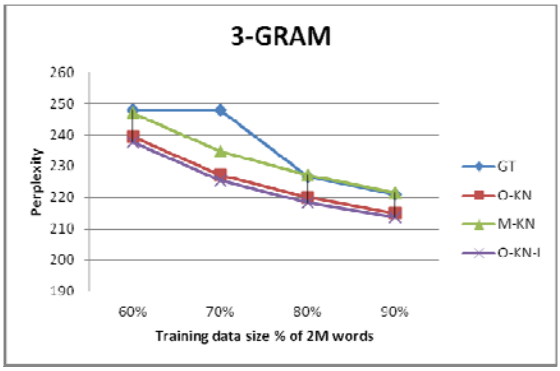


Figure 10. Both corpora, 3-gram LM

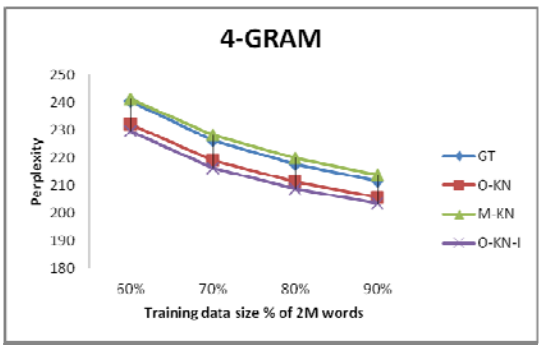


Figure 11. Both corpora, 4-gram LM

7. DISCUSSION

From the results shown in Figure 2 to Figure 9, we can see that the perplexity value decreases as the order of n-grams increases but then it becomes steady or saturated as it reaches higher orders for different smoothing techniques. The OOV rates for both the Helsinki Corpus and the Web Corpus decrease by similar factors as the size of the dictionary increases. (One thing to be cleared about the OOV rates is that our goal was not to compare them between the two corpora but to see the impact of dictionary sizes on the language model.) The language models created from the Helsinki corpus and the Web corpus are quite similar in terms of performance but they do differ slightly in terms of out of vocabulary words (OOV). The web corpus has more out of vocabulary words due to diversity of data coming from the web compared with the Helsinki corpus for which most of the data is taken from old newspapers which contain proper Swahili sentences. Interpolated original Kneser-Ney (O-KN-I) smoothing outperforms other smoothing techniques.

Based on different training data set sizes, the perplexity also decreases as data increases on different n-gram orders as shown in Figures 10 and 11. These results confirm those published earlier by Goodman in [6] based on comparison of different smoothing techniques against different n-gram orders and against varying sizes of training data.

8. CONCLUSION AND FUTURE WORK

This paper reported the work on how to collect Swahili content from the World Wide Web and prepare it to be used in building n-gram language models for various values of n, and subjected to various smoothing techniques. The process was accomplished with the assistance of Google search engine, a crawling tool and some Perl scripting for text processing. The process included both manual and automated tasks. The paper also shared our experience on performing text normalization, which was mainly done by constructing rules expressed in Perl. Finally, the preliminary language model results were presented which were similar to those reported in past work.

The web corpus data was not well cleaned up and the words were not tagged. It contained ungrammatical Swahili sentences, unwanted words and English words, which can give false results during real application such as speech recognition or machine translation. We are hoping in the future to use a morphological analyser to look for grammatical errors and spelling mistakes so that we can improve the quality of the corpus by reducing the OOV rate. The quality of the web corpus will be evaluated during the next phase of our research, when we apply it to machine translation. This will be compared with the performance of the Helsinki Corpus in the same task. For tagging the corpus, support vector machines or other classification algorithms could be used to perform this process more efficiently and effectively.

In text normalization, some conversion words such as (@-at, /-forward slash) were borrowed from the English dictionary. In the future, we are hoping to do more research, finding proper Swahili substitution words of these types of symbols. In Swahili-speaking countries, there are bodies responsible for language maintenance tasks, such as composing new words for modern phenomena. We hope to make contact with these and be able to tap into their resources. The language models created were not tested to check performance on real applications. The ultimate goal of this work is to perform such an evaluation by using these models in a machine translation system, which we intend to do in the last stage of this research.

.

12. REFERENCES

[1] M. Banko and E. Brill. 2001. "Scaling to very very large corpora for natural language disambiguation," In *Proceedings of the ACL*, pages 26–33, Toulouse, France,9–11 July.

[2] T. Brants, A.C. Popat, P. Xu, F.J. Och, and J. Dean, "Large Language Models in Machine Translation", in *Proc. EMNLP-CoNLL*, 2007, pp.858-867.

[3] G. De Pauw, G-M de Schryver & P.W. Wagacha. 2009. "A Corpus-based Survey of Four Electronic Swahili–English Bilingual Dictionaries," *Lexikos 19*: 340–352.

[4] T.T. Vu, D.T. Nguyen, L.C. Mai, and J. Hosom, "Vietnamese large vocabulary continuous speech recognition," in *Proc. INTERSPEECH*, 2005, pp.1689-1692.

[5] R. Sproat, A.W. Black, S.F. Chen, S. Kumar, M. Ostendorf, and C. Richards, "Normalization of non-standard words," presented at *Computer Speech & Language*, 2001, pp.287-333.

[6] J.T. Goodman, "A bit of progress in language modeling extended version," [R]. MSR-TR-2001-72 Technichal report, Microsoft Research, 2001.

[7] B. Wójtowics, *Investigationes Linguisticae*, vol. XI, Poznań, December 2004.

[8] A. Stolcke, "SRILM - An Extensible Language Modeling Toolkit," in *Proc. Intl. Conf. Spoken Language Processing*, Denver, Colorado, September 2002.

[9] N. Genserovskaya, "Text Normalisation for Irish Speech Synthesis," Final Year Project, School of Computer Science and Statistics, Trinity College, Dublin 2, Ireland, May 2007.