DEVELOPING NEW LANGUAGE TOOLS FOR MARYTTS: THE CASE OF LUXEMBOURGISH

Ingmar Steiner[1,2], *Sébastien Le Maguer*[1], *Judith Manzoni*[3], *Peter Gilles*[3], *Jürgen Trouvain*[1]
[1]*Saarland University*
[2]*DFKI GmbH*
[3]*University of Luxembourg*
steiner@coli.uni-saarland.de

Abstract: We present new methods and resources which have been used to create a text to speech (TTS) synthesis system for the Luxembourgish language. The system uses the MaryTTS platform, which is extended with new natural language processing (NLP) components. We designed and recorded a multilingual, phonetically balanced speech corpus, and used it to build a new Luxembourgish synthesis voice. All speech data and software has been published under an open-source license and is freely available online.

1 Introduction

With three official languages, the Grand Duchy of Luxembourg has approximately 400 000 speakers of Luxembourgish, the vast majority of which are also fluent in French and/or German. Moreover, Luxembourgish routinely imports words and expressions from both of those languages, and the result can be difficult to process from a monolingual perspective. With these facts in mind, it may be unsurprising that the speech technology industry has not invested into developing automatic speech recognition (ASR) or text to speech (TTS) products localized for Luxembourgish. While Adda-Decker et al. [1] have presented progress towards ASR in that language, no TTS synthesis was previously available for Luxembourgish.

Although various systems for English TTS synthesis cover at least two locales (British and American English), German synthesis is typically restricted to the variety spoken in Germany, although some efforts have been made to produce other dialects as well [2]. Although Luxembourgish is recognized as a sovereign language, it shares many characteristics with regional dialects of German along the Luxembourgish border. Meanwhile, methods and tools (e.g., [3]), have also become available to create synthesis systems of reasonable quality from limited linguistic resources and data.

In this paper, we describe the process to create natural language processing (NLP) components for Luxembourgish using the multilingual, open-source TTS platform MaryTTS [4].[1] We designed and recorded a speech corpus produced by a native speaker of Luxembourgish and German, who is also fluent in French. Finally, we built a synthesis voice from this corpus and integrated it into MaryTTS.

1.1 Current state and limitations

The process and tools required to extend MaryTTS with support for a new language were first developed for the Blizzard Challenge 2009 [5] and are described in more detail by Pammi et al. [6]. The workflow can be summarized as follows:

[1] `http://mary.dfki.de/`

1. A snapshot dump of Wikipedia in the new language is downloaded.
2. This dump is then uncompressed and split into individual pages, one per article.
3. Each page is converted from the MediaWiki markup format to plain text.
4. A list of unique words is created, sorted by frequency.

MaryTTS provides some shell scripts to automate these steps, but they suffer from numerous severe limitations. These include the reliance on a Linux environment, and administrative access to a running MySQL service. Accordingly, users who are unfamiliar with Linux (and command line interaction), and who have no experience with MySQL or a similar relational database management system can quickly run into any of numerous unhandled edge cases or other issues. These are exacerbated by the fact that all intermediate data is stored in the database itself and cannot by inspected at the file system level, making debugging and user support extremely tedious.

Incidentally, the software dependency used to extract the plain text from the MediaWiki markup[2] had not been actively maintained since 2009, and is also not free of problems.

5. The word list is phonetically transcribed by hand.
6. Letter to sound (LTS) rules are trained automatically.

Having extracted a list of the most frequent words in the new language, the user is required to manually create phonetic transcriptions, and then use them to train LTS rules, all within a Java GUI. Unfortunately, this GUI suffers from several usability and accessibility issues, which are difficult to debug and can even lead to data loss, not to mention user frustration. But of course the main issue is that the user is assumed to have some degree of phonetic expertise.

These steps are intended to produce resources which are needed for minimal NLP components to support the new language in MaryTTS.

7. Java code is developed for new language component classes.
8. Maven[3] is used to test and package the classes and resources into a new software component.

The requirement to develop new code and package it into a software component assumes that the user has a working understanding of software engineering for Java. Moreover, the user also needs familiarity with concepts specific to the Maven build tool, and there are several additional pitfalls, including a failure to correctly prepare a *ServiceLoader*, without which the new language component will simply not be "found" by a running MaryTTS instance.

9. Features are extracted from the Wikipedia sentences, including diphones and generic prosody symbols (derived from ToBI, [7]).
10. A list of prompt sentences is generated, maximizing the feature coverage via a greedy algorithm.

Assuming the minimal NLP components have been successfully created and loaded in a running MaryTTS instance, the proficient user can then run more shell scripts which connect MaryTTS with the MySQL database containing the Wikipedia sentences. These scripts rely on the output of the previous steps to predict the pronunciation of each sentence, and store it as features for a greedy algorithm [8], which aims to maximize diphone and prosodic coverage and finally, generate an optimal prompt list for recording.

[2] `org.wikipedia:mwdumper:1.16`
[3] `https://maven.apache.org/`

11. Prompt recording is done via yet another Java GUI, which displays the prompts and prepares them for voicebuilding.

Unfortunately, this recording tool (called *Redstart*) suffers from similar issues with usability and accessibility, and moreover is not free of bugs, which can even affect the recorded audio signal.

It seems plausible that the decision to rely fully on MySQL for handling the Wikipedia dump processing to generate the word and prompt lists was precipitated by the fact that the mwdumper library already took that approach. However, the MySQL connection adds a significant layer of provisioning overhead and obfuscation, which has caused problems for many users; in retrospect, it may not have been a sound design decision. Moreover, the apparent motivation to develop an end-to-end workflow using platform-independent Java GUIs is undermined by the reliance on Linux shell scripts for the text data processing, and obsoleted by the user's potential preference for alternative, specialized software.

1.2 Solutions

Over the past few years, MaryTTS development has increasingly embraced a new, agile build platform called Gradle.[4] Like Ant[5] and Maven, the previous build tools used for MaryTTS development, Gradle uses Java, but unlike them, it is extremely flexible and can be extended with new functionality and easily adapted to custom tasks. The first real proof of Gradle's flexibility and suitability for our purposes was delivered by using it to wrap and replace the VoiceImportTools used for creating new MaryTTS voices [9].

For the creation of LTS resources for new (and existing) languages, we can also use Gradle to radically streamline the process. The Wikipedia dump processing can be re-engineered by streaming the compressed dumps and converting them to plain text either in memory or via temporary files in MediaWiki format. The format conversion can nowadays be handled by many freely available libraries and tools; thanks to Gradle's flexibility, this task can also be delegated by executing externally provided programs such as Pandoc[6] from within the main build process.

The manual transcription of entries in the word list cannot be easily automated, but at least the user can freely choose which program to use; in the end, the list of transcribed words is simply a text file. Once prepared, this file is easy to convert to the LTS resources required for MaryTTS, using Gradle.

The basic Java source code required for the NLP components for the new language, is also easy to generate and build automatically, lifting this burden from the user's shoulders and minimizing the potential for numerous errors.

Finally, the actual prompt recording is best done via mature, dedicated digital audio workstation software, and there are numerous professional-quality programs available for all major operating systems, both commercial and free.

2 Luxembourgish NLP components for MaryTTS

In order to add support for Luxembourgish to MaryTTS, we were fortunate to have several assets at our disposal, including an existing pronunciation dictionary (see below), and access to a native speaker of Luxembourgish.

[4] `https://gradle.org/`
[5] `http://ant.apache.org/`
[6] `http://pandoc.org/`

We implemented two new processing components for text normalization and pronunciation prediction. Additional development can of course be done as needed in the future, but these two components comprise a functional NLP core required for Luxembourgish TTS.

Incidentally, we decided to write the source code using the Groovy language,[7] a lightweight but powerful alternative that is fully compatible, and can in fact be mixed, with Java, which allowed us to develop new code much more rapidly.

2.1 Text normalization

For the expansion of numeric content in Luxembourgish input text, we decided to rely on a third-party library instead of painstakingly developing new code (as had been done for German in the earliest version of MaryTTS). To this end, we declared a dependency on the ICU4J library,[8] and created a custom rule-based number format, which is used to spell out any numbers encountered, allowing them to be properly pronounced by MaryTTS.

2.2 Pronunciation modeling

In order to model Luxembourgish pronunciation, we first defined the set of allophones required for Luxembourgish TTS. With an initial set based on the inventory described by Gilles and Trouvain [10], we added several non-native sounds which would be required to correctly pronounce French, German, and English words (including [õ, ã], [yː, øː], and [θ, ð], respectively, among others).

Although we had begun our work by evaluating the existing workflow for new language support in MaryTTS (including the MySQL-based Wikipedia dump processing), we did not need to manually create phonetic transcriptions for the most frequent words. An existing website,[9] aimed at helping learners of Luxembourgish look up the meaning and pronunciation of a vocabulary of 6000 words, was initially used to create the word list required to generate the LTS resources for MaryTTS.

We quickly discovered, however, that many of the transcriptions provided had problems, so we pruned and manually corrected many of them. Ultimately, we replaced this resource with a manually transcribed word list after all, but only after the rest of the new language and voice building process had shown success.

As Luxembourgish texts routinely contain words from French (and – to a lesser degree – German), we added a custom rule which relies on the available French NLP components to predict the pronunciation of certain out-of-vocabulary words.

3 Creating a Luxembourgish synthesis voice

Of course, creating NLP components for Luxembourgish is not enough for actual TTS synthesis in that language; we needed an actual synthesis voice, built from suitable speech data. Accordingly, we proceeded to create a new, phonetically balanced multilingual synthesis corpus.

3.1 Prompt list

We designed a prompt list containing Luxembourgish, French, and German material suitable for TTS. Details of the prompt material are given in Table 1.

[7] http://www.groovy-lang.org/

[8] http://icu-project.org/

[9] http://www.6000wierder.lu/index.php

language		duration (min)
Luxembourgish	The Luxembourgish version of "The Northwind and the Sun"	0.5
	52 words spoken in isolation	0.5
	584 phonetically balanced sentences from the Luxembourgish Wikipedia	62
French	255 phonetically balanced sentences from the French Wikipedia	47
German	198 sentences from the phonetically balanced BITS corpus [11]	22

Table 1 – Prompt sets and durations. The orthography of the Luxembourgish Wikipedia had been manually corrected. The French Wikipedia prompts were taken from the prompt list used for the French MaryTTS voices created by Sathish Pammi at UPMC/ENST.

3.2 Recording procedure

In a single session (with several breaks), a female native speaker of Luxembourgish and German, who also speaks French fluently without an accent, was recorded in the studio at the Saarland University phonetics department. We used a close-talking microphone, as well as a second microphone mounted approximately 30 cm from the speaker's mouth. The audio was sampled at 48 kHz with 24 bits per sample.

The prompts were presented individually on a computer screen, with each one automatically triggering an inaudible beep that was recorded on a separate channel and used to automatically segment the session into prompts.

Overall, the recording session produced approximately 2 h 11 min of speech (see Table 1 for details).

3.3 Processing and segmentation

The recording session was stored as a single long take, which was automatically segmented into utterances based on the beeps triggered by the prompt display. These prompts were semi-automatically labeled, then each prompt was automatically phonetically segmented in a forced-alignment paradigm, using the MaryTTS wrapper for EHMM [12]. The segmentation for the Luxembourgish subset of the corpus was manually checked and corrected where necessary.

In order to allow this synthesis corpus to be efficiently used in a variety of scenarios, we losslessly compressed the audio and converted the metadata, following the FLAC+YAML convention [13].

3.4 Voicebuilding

Using the new voicebuilding toolchain implemented as a plugin for the Gradle platform [14], we built a Luxembourgish unit-selection voice from the corresponding corpus subset.

4 Conclusion

We have presented work that has produced a working TTS synthesis voice for the Luxembourgish language, using the MaryTTS platform. A live demo is available at `http://mary.dfki.de:59125/` (by selecting the voice "marylux"). The NLP components for Luxembourgish are

available online under the Lesser GPL license.[10] We have also released the synthesis voice, free for non-commercial use,[11] and the source data – a phonetically balanced, multilingual speech corpus – is publicly available under the same terms.[12] While the quality is not perfect, the open-source nature of this project's deliverables make it straightforward to improve the quality, and fix issues, as needed.

The development and integration of Luxembourgish support in MaryTTS coincided with several major, ongoing changes in the architecture and tooling of MaryTTS. A significant limitation still affecting the current version is the fact that each synthesis voice is by design associated with *exactly one* language; this made it challenging to efficiently use the full corpus recorded in Luxembourgish, French, and German together in single voice. However, we expect that moving to a more flexible feature model [14] will allow us to build truly multilingual synthesis voices in the future. Furthermore, we plan to apply incubating toolchains to also produce a statistical parametric voice, which would offer more flexible prosody control and a smaller memory footprint, and to evaluate the quality of the synthesis output.

We have shown that, and how, it is possible to adapt a TTS system such as MaryTTS to a new language. Most languages and language varieties are not considered for synthetic speech (just as for speech technology in general). Thus, this will be an emerging task for the future in this field. However, we have also shown that an adaptation to an existing TTS platform requires not only resources such as pronunciation dictionaries, texts, and sufficient data from a voice talent, but also a considerable know-how of technical adaptation. Therefore one task for the future will be to set up routines for adapting new languages, in this case for MaryTTS.

Acknowledgments

This work was partially funded by the University of Luxembourg. We are grateful to Maureen Tanuadji and Tristan Hamilton for their code contributions. We are also indebted to the EXPRESSION team at IRISA (Lannion, France) for their assistance with the prompt selection.

References

[1] ADDA-DECKER, M., L. LAMEL, and G. ADDA: *Speech alignment and recognition experiments for Luxembourgish*. In *4th Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU)*, pp. 53–60. St. Petersburg, 2014. URL `http://www.isca-speech.org/archive/sltu_2014/sl14_053.html`.

[2] PUCHER, M., D. SCHABUS, J. YAMAGISHI, F. NEUBARTH, and V. STROM: *Modeling and interpolation of Austrian German and Viennese dialect in HMM-based speech synthesis*. *Speech Communication*, 52(2), p. 164–179, 2010. doi:10.1016/j.specom.2009.09.004.

[3] SCHULTZ, T. and A. W. BLACK: *Rapid language adaptation tools and technologies for multilingual speech processing systems*. In *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Las Vegas, 2008. Tutorial T-3.

[4] SCHRÖDER, M. and J. TROUVAIN: *The German text-to-speech synthesis system MARY: A tool for research, development and teaching*. *International Journal of Speech Technology*, 6(4), pp. 365–377, 2003. doi:10.1023/A:1025708916924.

[10] `https://github.com/marytts/marytts`
[11] `https://github.com/marytts/voice-marylux-lb`
[12] `https://github.com/marytts/marylux-data`

[5] SCHRÖDER, M., S. PAMMI, and O. TÜRK: *Multilingual MARY TTS participation in the Blizzard Challenge 2009*. In *Blizzard Challenge Workshop*. Edinburgh, UK, 2009. URL http://festvox.org/blizzard/bc2009/dfki_Blizzard2009.pdf.

[6] PAMMI, S., M. CHARFUELAN, and M. SCHRÖDER: *Multilingual voice creation toolkit for the MARY TTS platform*. In *7th International Conference on Language Resources and Evaluation (LREC)*, pp. 3750–3756. Valetta, Malta, 2010. URL http://www.lrec-conf.org/proceedings/lrec2010/summaries/720.html.

[7] WIGHTMAN, C. W.: *ToBI or not ToBI?* In *Speech Prosody*, pp. 25–29. Aix-en-Provence, France, 2002. URL http://www.isca-speech.org/archive_open/sp2002/sp02_025.html.

[8] HUNECKE, A.: *Optimal Design of a Speech Database for Unit Selection Synthesis*. Master's thesis, Saarland University, 2006. Unpublished.

[9] SCHRÖDER, M., M. CHARFUELAN, S. PAMMI, and I. STEINER: *Open source voice creation toolkit for the MARY TTS platform*. In *Interspeech*, pp. 3253–3256. Florence, Italy, 2011. URL http://www.isca-speech.org/archive/interspeech_2011/i11_3253.html.

[10] GILLES, P. and J. TROUVAIN: *Luxembourgish (Illustrations of the IPA)*. *Journal of the International Phonetic Association*, 43(1), p. 67–74, 2013. doi:10.1017/S0025100312000278.

[11] ELLBOGEN, T., F. SCHIEL, and A. STEFFEN: *The BITS speech synthesis corpus for German*. In *4th International Conference on Language Resources and Evaluation (LREC)*. Lisbon, Portugal, 2004. URL http://www.lrec-conf.org/proceedings/lrec2004/summaries/72.htm.

[12] PRAHALLAD, K., A. W. BLACK, and R. MOSUR: *Sub-phonetic modeling for capturing pronunciation variations for conversational speech synthesis*. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 853–856. Toulouse, France, 2006. doi:10.1109/ICASSP.2006.1660155.

[13] STEINER, I.: *A DevOps manifesto for speech corpus management*. In *28th Conference on Electronic Speech Signal Processing (ESSV)*, pp. 160–166. Saarbrücken, Germany, 2017.

[14] LE MAGUER, S. and I. STEINER: *Uprooting MaryTTS: Agile processing and voicebuilding*. In *28th Conference on Electronic Speech Signal Processing (ESSV)*, pp. 152–159. Saaarbrücken, Germany, 2017.