Finite-state super transducers for compact language resource representation in edge voice-AI

Simon Dobrišek a, Žiga Golob b and Jerneja Žganec Gros b

aFaculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia; bAlpineon Research and Development Ltd., Ljubljana, Slovenia

ABSTRACT
Finite-state transducers have been proven to yield compact representations of pronunciation dictionaries used for grapheme-to-phoneme conversion in speech engines running on low-resource embedded platforms. However, for highly inflected languages even more efficient language resource reduction methods are needed. In the paper, we demonstrate that the size of finite-state transducers tends to decrease when the number of word forms in the modelled pronunciation dictionary reaches a certain threshold. Motivated by this finding, we propose and evaluate a new type of finite-state transducers, called 'finite-state super transducers', which allow for the representation of pronunciation dictionaries by a smaller number of states and transitions, thereby significantly reducing the size of the language resource representation in comparison to minimal deterministic final-state transducers by up to 25%. Further, we demonstrate that finite-state super transducers exhibit a generalization capability as they may accept and thereby phonetically transform even inflected word forms that had not been initially represented in the original pronunciation dictionary used for building the finite-state super transducer. This method is suitable for speech engines operating on platforms at the edge of an AI system with restricted memory capabilities and processing power, where efficient speech processing methods based on compact language resources must be implemented.

ARTICLE HISTORY
Received 13 December 2021
Accepted 11 June 2022

KEYWORDS
Speech synthesis; pronunciation dictionary; finite-state super transducers; automatic grapheme-to-phoneme conversion

Introduction
Consistent and accurate pronunciation of words is crucial for the user adoption of many speech engines, such as speech recognition and speech synthesis. To achieve the highest possible accuracy, the process of converting graphemic representation of words into their phonemic representation usually consists of two steps. First, a lookup into the available pronunciation dictionary is performed for a given word, and this typically provides its most accurate grapheme-to-phoneme conversion. However, many words may not be included in the available dictionary, as languages are constantly evolving and new words are emerging at a high rate. The phonetic transcription for such non-dictionary words needs to be determined by automatic grapheme-to-phoneme conversion methods, such as rule-based (Hahn et al., 2012; Jiampojamarn & Kondrak, 2010; Načinović et al., 2009) or machine learning (Lehnen et al., 2013; Taylor et al., 2021; Yolchuyeva et al., 2019) methods. Neural network models such as multilayer perceptrons, recurrent neural networks, and convolutional neural networks have also been used for grapheme-to-phoneme conversions (Yao & Zweig, 2015). In general, it has been shown that learning algorithms of neural network models can be improved by incorporating particle swarm optimization and a priori information into neural networks (Han et al., 2010; Han & Huang, 2008).

Both pronunciation dictionary lookup and in particular machine learning methods can be extremely memory consuming, especially for highly inflected languages, where pronunciation dictionaries often contain more than a million word forms. In some systems with limited memory resources, e.g. multilingual speech engines for embedded systems, the use of a large pronunciation lexicon and direct lookup methods is not appropriate. To overcome this limitation, memory-efficient representations of pronunciation dictionaries are needed, especially in edge-computing environments running on small embedded platforms, such as Raspberry Pi and others.

The most efficient methods for compact representations of pronunciation dictionaries include numbered automata (Daciuk & Weiss, 2011; Lucchesi & Kowaltowski, 1993), tries (Ristov, 2005), and finite-state transducers (FSTs) (Golob et al., 2012; Kiraz, 2001; Mehryar, 1994).
In this paper, a new type of finite-state transducers, called ‘finite-state super transducers’, is presented and discussed. The concept of finite-state super transducers (FSSTs) was first introduced in (Golob et al., 2016) and (Golob, 2014). Finite-state super transducers enable more compact representations of pronunciation dictionaries, and also grapheme-to-phoneme conversions of words that are not contained in the original dictionary (Golob et al., 2016). We prove this with novel experiments on the extended set of OptiLEX language resources.

**Speech technologies in edge voice-AI**

Voice communication has evolved into a pervasive method of managing and interacting with technological devices, from the start of use in smartphones and smart speakers to smart watches, cars, household appliances and other smart home applications (Wang et al., 2019), intelligent audio points in smart cities, and much more.

In the devices that support voice communication, voice processing is increasingly performed on the device or on the edge of the system. This also applies to the implementation of machine learning and AI-based algorithms to reduce latency and provide a better user experience. Other benefits offered by edge-AI include overcoming inherent cloud-based service limitations due to its reliance on connectivity, in addition to cost savings, as cloud API calls are not required (Beňo et al., 2021).

Protecting privacy and maintaining security are another key reason that edge voice-AI will go hand-in-hand with cloud-based voice solutions (Wang et al., 2020). Voice technologies will continue to offer a more individual experience as they become better at distinguishing and recognizing users based on their voice characteristics. Since this can be a privacy risk as it includes voice biometrics, edge voice-AI provides a solution. By not sending the user’s voice signal that can be used as a biometric identifier and that contains sensitive personal biometric data to the cloud, the service can completely avoid these privacy and security issues.

As edge voice-AI often runs on small embedded platforms with restricted memory capabilities and processing power (Hasanzadeh Mofrad & Mosse, 2018), efficient speech processing based on compact language resources, such as pronunciation lexicons, must be implemented.

**Language resources used in the experiments**

Consistent and accurate determination of word pronunciation is critical to the success of many speech technology applications. Most state-of-the-art speech engines performing automatic speech recognition (ASR) and text-to-speech synthesis (TTS) rely on lexicons, which contain pronunciation information for many words. To provide maximum coverage of the words, multiword expressions, or even phrases that commonly occur in a given application domain, application-specific words, or phrase pronunciations may be required, especially for application-specific proper nouns such as personal names or names of locations. Pronunciation lexicons for speech engines contain grapheme and allophone transcription of lexical words. The ‘x-sampa-SI-reduced’ phonetic alphabet, a subset of the X-SAMPA set as defined for Slovenian (Zemljak Jontes et al., 2002), is used in allophone transcriptions. An example of a pronunciation lexicon for a few Slovenian words is shown in Figure 1.

Pronunciation dictionaries of six languages from three different language groups were used to experiment with different representations of pronunciation dictionaries. For the Slavic language group, the Czech phonological corpus with 279,826 lexical entries (Bičan, 2022) and the Slovenian SI-PRON pronunciation dictionary (Žganec Groset al., 2006) with 1,239,410 lexical entries were used. The latter has been augmented with the initial version of the OptiLEX pronunciation dictionary (Žganec Gros et al., 2020) comprising 57,000 lexical entries, whereby all the duplicates had been eliminated (Golob et al., 2021). This extended ALP-SI pronunciation dictionary has been used in the second part of the experiments while investigating FSST models.

For the Germanic language group, we used the freely available CMUdict (Carnegie Mellon University, 2015) pronunciation dictionary for North American English consisting of 131,720 lexical entries and the transcribed German
part of the LibriVoxDe (Beilharz et al., 2020) with 264,049 unique entries. For the Romance language group, the Italian pronunciation dictionary FST-IT from the Festival Speech Synthesis toolkit (Black & Taylor, 1997) with 402,962 lexical entries and the Portuguese stress lexicon (Garcia, 2017) with 154,610 lexical entries were used.

Although the size of the selected six pronunciation dictionaries varies, the number of lemmas per language used in the experiments is comparable, as the selected Slavic and Romanic languages as well as German exhibit a richer inflectional paradigm than the English language.

**FST representations of pronunciation dictionaries**

An FST can be built to accept all the words from a given dictionary and to output their corresponding phonetic transcriptions. To ensure fast dictionary lookups and small size, the FST can be converted to a minimal deterministic FST (MDFST) using efficient determinization and minimization algorithms (Mehryar, 2000). The resulting MDFST has the smallest number of states and transitions among all equivalent FSTs (Mehryar, 1997). Figure 2 depicts an MDFST for a simple example pronunciation dictionary containing nine two-letter words \( aa, ab, ac, ba, bb, bc, ca, cb, \) and \( cc \). The simple example words are represented by all possible pairs of letters from an alphabet \( A \), containing three letters (graphemes), \( A = \{a, b, c\} \). For the sake of simplicity, the corresponding phonetic symbols (allophones) are the same as the graphemes from the alphabet \( A \), and all the phonetic transcriptions of the nine words are identical to their graphemic representations.

As can be noticed from Figure 2, the nine lexical entries can be represented with an MDFST containing three states and six transitions. We can now consider another case where the \( cccc \) entry is removed from the example dictionary. To represent such a dictionary, a more complex MDFST with more states and transitions is needed, as shown in Figure 3.

From the above examples, it can be deduced that in some cases pronunciation dictionaries containing more words can be represented by smaller and simpler MDFSTs.

In order to test the dependency between the size of the MDFST and the size of the corresponding pronunciation dictionaries, 12 sub-dictionaries of different sizes were created for each of the six pronunciation dictionaries introduced in Section 2. The lexical entries of the sub-dictionaries were randomly selected from the original dictionaries. An MDFST was then constructed for each of the 12 sub-dictionaries. Figure 4 shows the number of states for all the resulting MDFSTs in relation to the number of states of the MDFST representing the original entire dictionaries.

The MDFSTs with the highest number of states were expected to correspond to the original dictionaries containing all of the lexical entries. Interestingly, this is not the case for all of the six languages (Figure 4), which confirms the initial observations reported in (Golob et al., 2016). For languages from the Romanic in Slavic language groups, which are all heavily inflected, the size of the corresponding MDFST is up to 160% larger for smaller dictionaries, and it reaches the maximum number of states for dictionaries when covering 55% to 65% of the lexical entries from the original dictionary. Similar results for the highly inflected Portuguese language dictionary were reported in Lucchesi and Kowaltowski (1993).

In contrast, the size of the MDFST representing the English language dictionary is almost linearly dependent on the size of the dictionary. This phenomenon appears to correlate with the number of inflected word forms. A possible explanation could be that inflection rules in the highly inflected languages follow a similar pattern with few exceptions, so inflected forms can often be represented by a similar FST substructure for many different lemmas. If an inflected form is missing for a particular lemma, the remaining forms have to be represented by a different FST substructure, which may result in a larger final MDFST size.

To further test this hypothesis, we repeated the experiment with the augmented SI-PRON pronunciation dictionary, where the sub-dictionaries were created by
Figure 4. Normalized number of states of the created MDFSTs representing dictionaries for six different languages. For each language, 12 sub-dictionaries with different sizes were built, whereby their lexical entries were randomly chosen from the original full-size pronunciation dictionaries.

Figure 5. The number of MDFST states representing sub-dictionaries for Slovenian with different sizes, where first only lemmas were chosen randomly from the original dictionary, and later all the corresponding inflected forms were added.

As can be seen in Figure 5, the size of the MDFSTs representing the augmented SI-PRON sub-dictionaries for Slovenian, which were created as described, increases monotonically. It can be thus concluded that the missing inflected word forms significantly increase the complexity of the MDFST representing the dictionary.

Finite-state super transducers

The previous section demonstrated that missing inflected forms are actually increasing the size of MDFSTs representing the corresponding dictionaries. Therefore, it is beneficial for reducing the size of an MDFST to include all
the possible inflected forms for all the lemmas that are in the corresponding dictionary represented by the MDFST, even if they are very rare or not needed in a particular pronunciation dictionary implementation.

In the endeavour for minimizing a MDFST dictionary representation, another question arises. Can an MDFST representing a pronunciation dictionary be even smaller if it accepts also other words that are not part of the original source dictionary? We have found that this can be achieved by defining several specific rules for merging FST states. By analogy to supersets, we refer to the resulting FST as a finite-state super transducer (FSST), as initially proposed in Golob et al. (2016).

It should be noted that when new words are represented by such an extended FST, information about which words or lexical entries belong to the original dictionary and which have been added as the result of such an extension is lost.

Construction of FSSTs

The main idea behind the construction of FSSTs is to find new relevant words or strings that would allow additional merging of states. Instead of searching for such words, which is a very complex task, the problem can be solved by searching for relevant non/equivalent states that can be merged. For example, in the FST presented in Figure 3, states 1 and 2 could intuitively be merged. All input and output transitions of both states are part of a new merged state, only repetitions of identical output transitions need to be omitted. The resulting FST is an FSST, presented in Figure 2, which accepts the additional word cc.

We need to determine which states of a given MDFST are the best candidates to be merged. It is important to note that not all states can be merged, as certain merges could result in transforming the given MDFST into a non-deterministic FST, which is not desirable because such transducers can be slow and ambiguous in translating input words. After studying various options and limitations in transforming a MDFST into a FSST, we determined several rules for merging states that preserve the determinism of the obtained transducer. Two states, which satisfy these rules, are denoted as mergeable states. The rules for merging states (Golob et al., 2016) are:

- Mergeable states do not have output transitions with the same input symbols and different output symbols;
- Mergeable states do not have output transitions with the same input and output symbols and different target states;
- If one of the two mergeable states is final, both states do not have output transitions with input symbol that is an empty character ϵ.

The above rules are stricter than necessary to preserve the determinism of the final FSST and are also very easily verified. The goal of this paper is not to aim at building the smallest possible FSST, but to demonstrate the potential of FSSTs.

To build an FSST from an MDFST, we defined an algorithm that searches for mergeable states by checking all possible combinations of two states of the input MDFST. Each time a pair of mergeable states is found, they are merged into a single state. Since merging affects the other states that have already been verified against the rules mentioned above, several iterations are normally needed until no new mergeable states have been found.

In our experiments, which we repeated on the extended language resources described in Section 2, we observed that the final size of an FSST depends on the order of state merging. The implementation size or memory footprint of an FST mostly depends on the number of transitions and less on the number of states (Golob, 2014). We experimentally proved the findings in Golob et al. (2016) that the smallest final number of transitions is obtained only if the states with the highest number of identical transitions are merged in the first iterations, since the repetitions of all the identical transitions can be immediately removed from the transducer (Golob et al., 2021). In order to lower the number of the additional words that are accepted by the resulting FSST, those mergeable states whose merging did not decrease the number of transitions are not merged.

Experimental results

The experiments were conducted using the methodology proposed in Golob et al. (2016). Initially, two MDFSTs were built for each available pronunciation dictionary, using the open-source toolkit OpenFST (Allauzen et al., 2007). For the second type of MDFSTs, denoted as MDFST-2, the output strings of transitions were constrained to the length 1 (in contrast to the first type denoted as MDFST-1, which does not have this restriction). The second type of MDFSTs normally has more states and transitions than the first type. However, it exhibits a simpler implementation structure, which results in a smaller implementation size.

FSSTs were then built from all the MDFSTs using the rules presented in the previous subsection. The results are presented in Tables 1 and 2.

Non-dictionary words

A system for converting graphemes to phonemes usually consists of two steps. First, it checks whether the input
Table 1. A comparison of the number of states and transitions between MDFSTs and FSSTs obtained with the first type of FSTs.

| Dictionary | MDFST-1 States | MDFST-1 Transitions | FSST States | FSST Reduction | FSST Transitions | FSST Reduction |
|------------|----------------|---------------------|-------------|----------------|-----------------|----------------|
| ALP-SI     | 69,524         | 239,852             | 56,120      | 19.3%          | 210,665         | 12.2%          |
| CMU-US     | 76,035         | 184,232             | 60,477      | 20.5%          | 164,235         | 10.9%          |
| FST-IT     | 57,323         | 172,210             | 44,563      | 22.3%          | 148,657         | 13.7%          |

Table 2. A comparison of the number of states and transitions between MDFSTs and FSSTs obtained with the second type of FSTs.

| Dictionary | MDFST-2 States | MDFST-2 Transitions | FSST States | FSST Reduction | FSST Transitions | FSST Reduction |
|------------|----------------|---------------------|-------------|----------------|-----------------|----------------|
| ALP-SI     | 224,123        | 533,561             | 170,832     | 23.8%          | 426,225         | 20.1%          |
| CMU-US     | 185,053        | 307,027             | 153,746     | 16.9%          | 269,677         | 12.2%          |
| FST-IT     | 157,379        | 318,805             | 124,717     | 20.8%          | 260,447         | 18.3%          |

word is contained in the original pronunciation dictionary. If this is not the case, a phonetic transcription of the input word needs to be determined using appropriate statistical or machine learning methods. In contrast, if an FSST is used to represent a pronunciation dictionary and it accepts the input word, then it is not possible to determine whether the output phonemic transcription is correct, since an FSST has the ability to generalize and may also accept words that are not part of the original dictionary, and these non-dictionary words may have incorrect phonemic transcriptions.

In order to evaluate this error, the augmented Slovenian pronunciation dictionary described in Section 2 was divided into a training and a test set. The training set contained 90% of the lexical entries from the original dictionary, which were randomly selected. The remaining entries represented the test set. An MDFST-2 and an FSST were then created from the training set. The results for the number of states and transitions are presented in Table 3.

The results in Table 3 show that the reduction of the number of FSST states and transitions is higher when not all inflected forms are contained in the dictionary. The number of states and transitions of the obtained FSST is even lower than that of the MDFST representing all the inflected forms.

In the first experiment, the FSST was built from the training set and then words from the test set, which represented non-dictionary words, were used as its input. Accepted and rejected words were then enumerated and counted, and the accuracy of the phonetic transcriptions was checked on the output. The results of this experiment are shown in Figure 6.

Interestingly, similarly to (Golob et al., 2016), only 8.24% of non-dictionary words were not accepted by the FSST and for only 5.83% of the accepted words the output phonetic transcription was incorrect. Overall, this results in 85.93% phonetic transcription accuracy for unknown input word forms. This outperforms comparable grapheme-to-phoneme conversion results reported for the Slovenian language, where phonetic transcription accuracies of up to 83% (Šef et al., 2002) by using machine learning methods have been obtained, and the recently reported 84% phonetic transcription accuracy of the input word forms (Križaj et al., 2022) by using the state-of-the-art DeepPhonemizer Transformer-based models, for which we provided a pretrained model and made it freely available (Križaj et al., Deep Models for Slovenian Grapheme-to-Phoneme Conversion, 2022).

Nevertheless, it should be noted that since test entries were chosen randomly from the original dictionary, the inflected forms that belong to the same lemma could be present both in the training as well as in the test dictionary. In this way, both dictionaries were partially similar, even though the word forms were different.

Therefore, we repeated the experiment by first choosing lemmas randomly and then only the inflected forms of
the chosen lemmas were added to the training dictionary. Figure 7 shows the results for this second experiment where due to missing lemmas only 42.33% of the words were accepted by the FSST. Among the accepted words, the phonetic transcriptions were still correct for 84.59% of the words, which is comparable to the already mentioned 84% accuracy of the DeepPhonemizer model for grapheme-to-phoneme transcription for Slovenian (Križaj et al., 2022), which anyway need to be applied to phonetically transcribe the word forms rejected by the FSST. The mentioned DeepPhonemizer model was trained in a similar way as the FSST with separate lemmas in the training and the test datasets, and the reported results are thus adequately comparable.

Still, it is important to stress that this experiment represents an extreme case where entire lemmas along with their word forms are missing in the original pronunciation dictionary used for building FSST models. This is usually not the case since in a given language the word forms that are represented by pronunciation dictionary lexical entries are typically harvested from large text corpora using word frequency analysis. Thereby, most of the lemmas in a given language are represented in the lexicon by at least partially incomplete inflectional paradigms whereby less frequent word forms may be missing, which coincides with the first experimental setup and results presented in Figure 6.

Discussion

The implementation size of FSTs representing pronunciation dictionaries depends mainly on the number of transitions, as these carry most of the information. The number of transitions is also usually several times larger than the number of states.

The results in Tables 1–3 show that using the proposed FSSTs, their size in terms of the number of states and transitions can be reduced up to 25%, which confirms the initial findings of a previous study (Golob et al., 2016).

The experiments were conducted on language resources in six languages each belonging to a different language group. The highest reduction rate was observed for the highly inflected Slovenian language when some inflected forms were missing in the original dictionary. As can be seen in Figure 4, non-included or missing inflected forms can significantly increase the size of an MDFST. If missing inflected forms are not added to the dictionary, an FSST can always be built from an MDFST. In this way, most of the inflected forms are automatically added to the pronunciations dictionary represented by an FSST.

The proportion of the accepted non-dictionary words with correct phonetic transcriptions is surprisingly high for FSSTs. For rejected words, the phonetic transcription can be determined using other grapheme-to-phoneme approaches. This is particularly important when there is a high probability that the input word will be rejected by an FSST. This mainly occurs when a word belongs to a new lemma for which inflected forms are not represented in the original pronunciation dictionary.

Conclusions

The presented results, obtained on an augmented data set confirm that by using FSSTs as initially introduced in (Golob et al., 2016), their size in terms of number of states and transitions can be reduced up to 25%. The highest reduction rate was observed for the highly inflected Slovenian language when some inflected forms were missing from the original dictionary. Missing inflected forms can significantly increase the size of an MDFST. When missing inflected forms are not added to the dictionary, an FSST can always be built from an MDFST, and in this way, most of the missing inflected forms are automatically represented by an FSST.

As a conclusion, FSSTs can be used as very compact and memory efficient computational models for grapheme-to-allophone conversions in speech engines. Compared to MDFSTs, their implementation size can be reduced by up to 20%. The reduction seems to be much higher for highly inflected languages, when not all the inflected forms are originally contained in the represented dictionary.

An important finding of this study is that a large proportion of the out-of-dictionary words that are accepted by an FSST are also correctly converted to their phonetic transcriptions. For the words rejected by an FSST, the phonetic transcription can always be determined using other grapheme-to-phoneme conversion methods. In our experiments, the phonetic transcriptions of the accepted non-dictionary words were correctly determined for up to 86% of the words. This may be particularly useful for implementation in speech engines.
designed for systems with limited memory resources (e.g. in edge voice-AI engines running on small embedded platforms), where the size of language resources of the speech engines needs to be radically reduced. Finally, there is the question of how the proposed FSST language resource representation methods are best implemented, particularly in the context of speech synthesis and speech recognition engines running in low-resource platform setups. Our future research will focus in this direction in order to drive the technology adoption of the proposed methodology.

Another direction of future research will be concerned with the extension of our experiments to additional languages with a focus on language groups sharing rich inflectional paradigms, which typically result in extensive pronunciation dictionaries, with a particular interest in other related southern Slavic languages.

**Disclosure statement**

No potential conflict of interest was reported by the author(s).

**Funding**

This work was funded by the Slovenian Research Agency in scope of the applied research project L7-9406 OptiLEX and the research programme P2-0250(C) Metrology and Biometric Systems.

**ORCID**

Simon Dobrišek [http://orcid.org/0000-0002-9130-0345](http://orcid.org/0000-0002-9130-0345)

Jerneja Žganec Gros [http://orcid.org/0000-0001-5011-8486](http://orcid.org/0000-0001-5011-8486)

**References**

Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., & Mohri, M. (2007). OpenFST: A general and efficient weighted finite-state transducer library. In *Implementation and Application of Automata* (Vol. 4783, pp. 11–23). Springer.

Beilharz, B., Sun, X., Karimova, S., & Riezler, S. (2020). LibriVoxDeEn: A corpus for German-English speech translation and German speech recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference* (pp. 3590–3594). European Language Resources Association. [https://aclanthology.org/2020.lrec-1.1441](https://aclanthology.org/2020.lrec-1.1441)

Beño, L., Pribiš, R., & Drahoš, P. (2021). Edge container for speech recognition. *Electronics*, 10(19), 2420. [https://doi.org/10.3390/electronics10192420](https://doi.org/10.3390/electronics10192420)

Bičan, A. (2022, March). Phonological corpus of Czech. Retrieved May 9, 2022, from Czech Language Institute, Czech Academy of Sciences: [https://ujc.avcr.cz/phword/](https://ujc.avcr.cz/phword/)

Black, A. W., & Taylor, P. A. (1997). *The festival speech synthesis system: System documentation*. Human Communication Research Centre, University of Edinburgh.

Carnegie Mellon University. (2015, February). *Carnegie Mellon pronouncing dictionary*. Retrieved November 13, 2021, from GitHub: [https://github.com/cmusphinx/cmudict](https://github.com/cmusphinx/cmudict)

Daciuk, J., & Weiss, D. (2011). Smaller representation of finite state automata. In *Implementation and Application of Automata: CIAA’11 Proceedings of the 16th International Conference on Implementation and Application of Automata* (pp. 118–129).

Garcia, G. D. (2017, March). Portuguese stress lexicon. [https://doi.org/10.17605/OSF.IO/XMZBU](https://doi.org/10.17605/OSF.IO/XMZBU)

Golob, Ž. (2014). Reducing redundancy of finite-state transducers in automatic speech synthesis for embedded systems [Doctoral dissertation, University of Ljubljana, Faculty of Electrical Engineering]. (F. o. University of Ljubljana, Ed.).

Golob, Z, Vesnicer, B., Žganec, M., Štruc, V., Dobrišek, S., & Žganec Gros, J. (2021). Compact finite-state super transducers for grapheme-to-phoneme conversion in highly inflected languages. In D. Huang, K. Jo, J. Li, V. Gribova, & V. Bevilacqua (Eds.), *Intelligent Computing Theories and Application. ICIC 2021. Lecture Notes in Computer Science*. 12836. Springer. [https://doi.org/10.1007/978-3-030-84522-3_50](https://doi.org/10.1007/978-3-030-84522-3_50)

Golob, Z, Žganec Gros, J., Štruc, V., Mihelič, F., & Dobrišek, S. (2016). A composition algorithm of compact finite-state super transducers for grapheme-to-phoneme conversion. In *Text, Speech and Dialogue: Proceedings, 19th International Conference, TSD 2016* (pp. 375–382). Springer.

Han, F., & Huang, D.-S. (2008). A new constrained learning algorithm for function approximation by encoding a priori information into feedforward neural networks. *Neural Computing and Applications*, 17(5), 433–439. [https://doi.org/10.1007/s00521-007-0135-5](https://doi.org/10.1007/s00521-007-0135-5)

Han, F., Ling, Q.-H., & Huang, D.-S. (2010). An improved approximation approach incorporating particle swarm optimization and a priori information into neural networks. *Neural Computing and Applications*, 19(2), 255–261. [https://doi.org/10.1007/s00521-009-0274-y](https://doi.org/10.1007/s00521-009-0274-y)

Hasanzadeh Mofrad, M., & Mosse, D. (2018, October). Speech recognition and voice separation for the internet of things. In *IOT’18: Proceedings of the 8th International Conference on the Internet of Things* (pp. 1–8). [https://doi.org/10.1145/3277593.3277610](https://doi.org/10.1145/3277593.3277610)

Jiampojamarn, S., & Kondrak, G. (2010). Letter-phoneme alignment: An exploration. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 780–788). Association for Computational Linguistics.

Kirázi, G. A. (2001). Compressed storage of sparse finite-state transducers. In *Automata Implementation, 4th International Workshop on Implementing Automata, WIA’99* (pp. 109–121). Springer-Verlag.

Križaj, J., Dobrišek, S., Mihelič, A., & Žganec Gros, J. (2022). Recent advances in Slovenian automatic grapheme-to-phoneme conversion. In D. Fišer & T. Erjavec (Eds.). *Submitted for Publication in the Proceedings of the Language Technologies and Digital Humanities Conference 2022*. Ljubljana University Press, Faculty of Arts.
Lehnen, P., Allauzen, A., Lavergne, T., Yvon, F., Hahn, S., & Ney, H. (2013). Structure learning in hidden conditional random fields for grapheme-to-phoneme conversion. *Proceedings of Interspeech* 2013 (pp. 2326–2330). ISCA.

Lucchesi, C. L., & Kowaltowski, T. (1993). Applications of finite automata representing large vocabularies. *Software-Practice & Experience*, 23(1), 15–30. https://doi.org/10.1002/spe.4380-230103

Mehryar, M. (1994). Compact representations by finite-state transducers. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics* (pp. 204–209). Association for Computational Linguistics.

Mehryar, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2), 269–311. https://aclanthology.org/J97-2003/

Mehryar, M. (2000). Minimization algorithms for sequential transducers. *Theoretical Computer Science*, 234(1–2), 177–201. https://doi.org/10.1016/S0304-3975(98)00115-7

Načinović, L., Pobar, M., Ipšić, I., & Martinčić-Ipšić, S. (2009). Grapheme-to-phoneme conversion for Croatian speech synthesis. In N. Bogunović & S. Ribarić (Eds.), *32nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO 2009): Proceedings* (Vol. 3, pp. 318–323). Croatian Society for Information and Communication Technology, Electronics and Microelectronics.

Ristov, S. (2005). LZ trie and dictionary compression. *Software-Practice & Experience*, 35(5), 445–465. https://doi.org/10.1002/spe.643

Šef, T., Škrjanc, M., & Gams, M. (2002). Automatic lexical stress assignment of unknown words for highly inflected Slovenian language. In K. I. Sojka (Ed.), *Text, Speech and Dialogue*: TSD 2002. *Lecture Notes in Computer Science* (Vol. 2448, pp. 165–172). Springer.

Taylor, J., Le Maguer, S., & Richmond, K. (2021). Liaison and pronunciation learning in end-to-end text-to-speech in French. In *Proceedings of 11th ISCA Speech Synthesis Workshop (SSW 11)* (pp. 195–199). https://doi.org/10.21437/SSW.2021-34

Wang, P., Lu, X., Sun, H., & Lv, W. (2019). Application of speech recognition technology in IoT smart home. In *2019 IEEE 3rd Advanced Information Management, Communications, Electronic and Automation Control Conference (IMCEC)* (pp. 1264–1267). https://doi.org/10.1109/IMCEC46724.2019.8984175

Yao, K., & Zweig, G. (2015). Sequence-to-sequence neural net models for grapheme-to-phoneme conversion. *Proceedings of INTERSPEECH* 2015 (pp. 3330–3334). ISCA.

Yolchuyeva, S., Németh, G., & Gyires-Tóth, B. (2019). Transformer based grapheme-to-phoneme conversion. *Proceedings of Interspeech 2019* (pp. 2095–2099). ISCA. https://doi.org/10.21437/Interspeech.2019-1954

Zemljak Jontes, M., Kačič, Z., Dobrišek, S., Gros, J., & Weiss, P. (2002). Računalniški simbolni fonetični zapis slovenske govore. *Slovenščina in računalniki* (pp. 159–169). https://dk.um.si/IzpisGradiva.php?lang=eng&id=53130

Žganec Gros, J., Cvetko Orešnik, V., & Jakopin, P. (2006). SI-PRON pronunciation lexicon: A new language resource for Slovenian. *Informatica*, 30(4), 447–452.

Žganec Gros, J., Golob, Ž., & Dobrišek, S. (2020). Učinkovita predstavitev slovanskih jezikovnih virov pri govornih tehnologijah. In *Proceedings of the Human-Computer Interaction in Information Society Conference, Information Society – IS 2020* (Vol. H, pp. 45–48).