# Deep Learning's Impact on Speech Synthesis for Mobile Devices

Ronit Chougule[1] and Akshay Kamkhalia[2]

[1] UG Student, School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT WORLD PEACE UNIVERSITY, Pune-411038, India
1032200841@mitwpu.edu.in

[2] Department of Computer Science, University of Mumbai, Mumbai, India
akshaykamkhalia2@gmail.com

**Abstract.** In recent years, speech synthesis has become an integral part of speech-based human-machine interfaces on mobile devices, sparking significant research interest. While traditional approaches like formant-based synthesis, unit-selection synthesis, and Hidden Markov Model (HMM)-based synthesis have been widely used, HMM-based synthesis, a type of Statistical Parametric Speech Synthesis (SPSS), has gained popularity due to its compact model size and flexibility in adjusting voice characteristics. However, the synthetic speech produced by HMM-based synthesis often sounds muffled due to over-smoothing, indicating the need for improvement. Implementing speech synthesis on mobile devices also presents challenges, such as limited memory resources, processing capacity, and real-time responsiveness. To tackle these challenges, researchers have investigated the use of deep learning models to improve voice quality, enhance prediction performance, and optimize embedded implementation without compromising voice quality in HMM-based synthesis systems. Deep learning models have also been applied in different stages of conventional SPSS to significantly reduce the memory footprint. These findings suggest that deep learning has a significant impact on speech synthesis for mobile devices, with the potential for further improvements in voice quality and efficient embedded implementation. Thus, this area presents a promising research direction for the future.

**Keywords:** Deep Learning, Speech Synthesis, Hidden Markov Model.

## 1 Introduction

VPAs like Siri, Cortana, or Google Now start having a huge impact on the way of interacting with electronic devices like smartphones or notebooks. Up to now, the VPA helps with rather simple tasks such as search queries, starting phone calls, or setting a clock, but according to a recent survey from the IT research firm, Gartner [4], this will change in the near future. With Facebook Messenger it is already possible to make purchases or to order an Uber car and new use cases are expected soon. The survey also states that through the vast increase of devices in the scope of the IoT the way of interacting with machines will go towards minimal or zero touch. Instead of interacting through common touch displays or buttons, the user simply speaks to the device like to another person. To enable this, both speech recognition and speech synthesis are essential technologies.

This paper will exclusively concentrate on the aspect of speech synthesis. A widely spread technique to synthesize human speech from a given text or linguistic description is SPSS, also referred to as SPSG [13]. According to Black *et al.*, the most preferred instance of SPSS is the HMM-based approach [5]. The authors have shown that it has several advantages over its predecessor, the concatenative speech synthesis, for example, the flexibility in changing voice characteristics or the smaller memory footprint. However, the quality of the generated speech still has potential for improvement. Due to over-smoothing the voice sounds muffled in comparison to natural speech.

This is where recent achievements in deep learning come in. Deep learning is usually referred to as a class of machine learning techniques that achieve tasks like feature extraction or pattern analysis by using many connected layers of non-linear information processing [13, 9]. Since 2006, advances in the training algorithms of DNN have enabled the field of deep learning applications to emerge [6]. Most machine learning models until then had used shallow structures, like HMM, GMM, CRF, or SVM. In these structures, only one layer is responsible for generating features out of the raw input signals. While achieving quite good results with rather simple problems, they reach their limit when it comes to more complex tasks like processing human language or natural images [9]. In the tutorial survey [9], the author also states four different approaches to improve speech synthesis through deep learning models, where three deal with HMM-based speech synthesis. One of those three approaches is described in [17], where the authors implemented a part of the speech synthesis system by using a DNN and observed an improved performance in predicting output features. In [11], a comprehensive investigation was carried out to examine the impact of deploying a DNN on various components of an HMM network. The research yielded significant advancements in the naturalness the synthesized speech as one of its key outcomes.

For implementing speech synthesis on resource-constrained devices like smartphones or tablets, SPSS is considered the best solution due to the trade-off between voice quality and acceptable footprint size [15]. Since the computational costs of SPSS are often high, some optimization steps are conducted [15] to make an HMM-based instance of SPSS more suitable for mobile devices. These steps include reducing the size of the decision trees and introducing streaming synthesis. Going one step further, in [6] an approach to adapt HMM-based speech synthesis for mobile devices by using a deep learning model, an autoencoder, is employed. Four tasks (syllabification, phonetic transcription, part-of-speech tagging, and lexical stress prediction) are examined and tested with the use of this deep learning model. As a result, the authors highlight hugely reduced model sizes and a very close performance to cutting-edge models. This shows that deep learning models for speech synthesis on resource-constrained systems are reasonable, not only to improve performance and voice quality but also towards the independence of online databases for speech synthesis.

The remaining parts of the paper are: Section 2, to explain why speech synthesis is a useful technology. Then it describes the conventional approach without deep learning models for speech synthesis and gives an overview of the advantages and drawbacks of the used models and techniques. This is followed by a brief explanation of how speech synthesis can be modeled on HMM, which is a popular type of SPSS. The paper [5] has been chosen as a typically cited reference. Thereafter, two possibilities for how HMM-based speech synthesis can be improved by deploying deep learning models are characterized. For this purpose, the papers [17] and [11] are reviewed. In Section 4, the motivation for why speech synthesis is important on mobile devices is given, followed by two examples of how speech synthesis can be implemented on a mobile device, one without [15] and one with deep learning models [6]. Section 5 discusses the results obtained from the methods illustrated in Section 4. Section 6 briefs about the study's core focus and ponders what directions can sustain later research.

## 2 Conventional Speech Synthesis

### 2.1 Motivation and Approaches

Speech synthesis has emerged over the last ten years due to a vast contribution by the global community of researchers and the increasing computational power for data processing. Its quality and naturalness have increased steadily and different approaches have been developed so far [14]. Typical applications like navigation systems in cars or telephone-based dialogue systems are nowadays widely established. But also, as a reading aid for visually impaired people [1] or as in the case of the famous scientist Stephen Hawking, who has been using a synthesized voice to communicate since 1997 [3], speech synthesis has proven to be very useful. Another very interesting application of speech synthesis is shown in [10]. Here, the author proposed to introduce synthetic speech as a means of communication between pilots, since there have been many accidents due to misunderstandings in radio-based communication.

According to [12], speech synthesis can be divided into three types: Canned speech, CTS, and Text-to-Speech (TTS). Canned speech more or less is the playback of prerecorded spoken sentences or words, both with no or very little adjustments. Typical examples are the announcements at train stations. Because of the high effort of recording everything (almost) exactly as it is played back, this approach is limited to a few simple applications. With the second type, CTS, the waveform is generated out of a linguistic description without any information about the respective text. In this way, no natural language processing is required, but CTS has not made any important impact yet. The last and most promising type is TTS. A TTS system consists of a NLP part, where the text is analyzed and the word and sentence structure and accents are extracted. Subsequently, these accents are employed to generate the prosodic aspects of the provided text. The phonetic representations, along with the incorporated prosodic information, are then seamlessly combined into a continuous stream of signal parameters. The last task, speech generation, uses this stream to generate the respective waveform. This function block can be implemented in different ways. In [12], three general approaches are named as follows: Parametric Speech Synthesis, Concatenative Speech Synthesis, and SPSS (HMM-based synthesis). The methods in brackets are the respective implementations, which are most commonly used.

The formant-based synthesis is the oldest approach. To generate a voice waveform, an excitation signal is fed into multiple formant filters, which describe the characteristics of the human vocal tract. The output of the filters then forms the voice waveform. Unlike other techniques that rely on recorded speech, this approach exclusively models the human vocal tract to generate the synthesized voice, eliminating the need for any pre-recorded audio data. While the generated voice quality of formant-based synthesizers is relatively lower compared to other techniques, they offer the advantage of having a smaller footprint. Additionally, modifying the voice characteristics of formant-based synthesizers can be easily achieved by adjusting the filter parameters in [12]. With the development of a concatenative speech synthesizer, the quality of the generated speech improved tremendously. Very similar to CTS prerecorded speech is used as reference. The recorded speech is divided into units and these units are then strung together to form the new speech signal according to a given text. Hereby, the chosen size of the units determines both the footprint size and the voice quality. With larger units, a higher voice quality can be achieved, but this also results in a much larger database. The challenge in unit selection is to ensure that the transition between the units is as natural as possible [12].

## 2.2 HMM-based Speech Synthesis

In this section, the HMM-based speech synthesis which is an instance of SPSS and the most recent approach will be described further. Additionally, both advantages and drawbacks compared to unit-selection synthesis will be highlighted. Therefore, the work of Black *et al.* [5] will be taken as a reference since this work is widely accepted and commonly cited when dealing with this topic.

The excellence of unit-selection synthesis is intimately tied to the excellence of the prerecorded speech employed. But even with a database of excellent quality, sporadic errors can still not be avoided totally. If a specific phonetic or prosodic part of a generated sentence is not well represented in the database the output quality of this sentence suffers immensely. To try to avoid this, a huge effort in specifically designing the database for the required application can be performed, but still, there is no guarantee that such bad joins will not happen. In addition, the fact that in unit selection no or only very little adaptions of the voice, characteristics are possible without an enormous increase of the database size, the ambition towards seamless speech synthesis leads towards the HMM-based approach. Here, a statistic representation of some sets of speech segments is used to generate arbitrary synthetic speech. In Figure 1, the structure of a typical HMM-based synthesizer is shown. The entire system can be segmented into two components: training and synthesis. Bridging these two components is a collection of context-dependent factors HMM.

In the training part spectrum and excitation parameters of the recorded speech are used to generate acoustic models represented by the HMM. Therefore, phonetic, linguistic, and prosodic parameters are considered. In comparison to a unit-selection system, the large speech database is only needed in the training part. During the synthesis phase, the input text undergoes a transformation into a sequence of context-related parameters, providing essential information for the synthesis process. According to this sequence, the respective HMMs are concatenated in order to form an utterance HMM. Then, after determining the state durations of the HMM, a sequence of coefficients is created, which finally is used to construct the speech waveform using a specific filter (*e.g.* an MLSA filter). In contrast to unit-selection synthesis, this approach suffers from drawbacks that contribute to the diminished quality of the synthesized speech. Several factors, namely the vocoder, modeling precision, and the phenomenon of excessive smoothing are responsible for this degradation. However, there are some essential advantages, which make the HMM-based approach a competitive alternative to unit-selection synthesis.

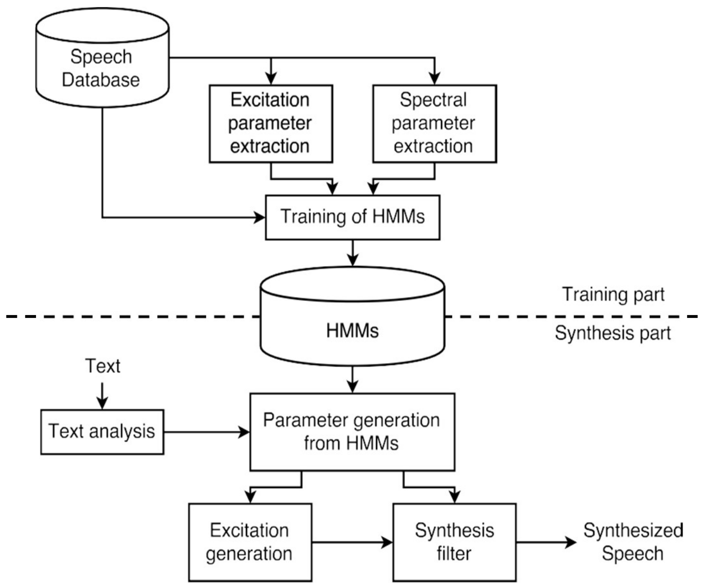

**Fig. 1.** Function blocks of HMM-based synthesis [5].

First, the voice characteristics can be modified without much effort. Thus, the implementation of different languages and the realization of different speaking styles with emotional emphasis is possible. Second, these aspects require a much smaller database than in unit-selection synthesis, since only a statistic representation of speech segments rather than raw speech data is stored. Finally, the quality of the synthesized speech of an HMM-based system is much more robust and free of sporadic flaws. Table 1 compares the three discussed techniques, highlighting their most prominent advantages and disadvantages.

**Table 1.** Comparison of speech generation methods [12,5].

| Technique | Advantages | Drawbacks |
|---|---|---|
| Formant-based | Very small footprint | Very artificial and metallic voice |
| Unit-selection | Very high voice quality possible | Large database required |
| HMM-based | Adjustable voice and small footprint | Voice sounds muffled |

## 3 SPSS with Deep Learning Models

Although the usage of SPSS has brought many advantages over unit-selection synthesis as shown in the previous section, the generated voice is still not as natural as desired. Therefore, deep learning models recently have been used to further improve SPSS. Since DNNs have proven to be very effective in speech recognition, they have found their way into speech synthesis, too [11]. At the beginning of this section, one specific approach to introduce deep learning models into an HMM-based speech synthesis system is explained further by using [17] as a reference. Then, some alternative techniques for enhancing the quality of speech synthesis through the deployment of deep learning models are outlined, as shown in the work of Hashimoto *et al.* [11].

### 3.1 One Specific Approach for Improvement

We highlighted the underlying factors responsible for the quality limitations observed in HMM-based speech synthesis. These factors encompass the vocoder, the precision of acoustic models, and the presence of the over-smoothing effect. Zen *et al.* [17] suggest a specific approach to eliminate one of these causes, in particular, the accuracy of acoustic models by allocating this task to a DNN.

In conventional HMM-based systems, the mapping between context features (phonetic and linguistic properties) and speech parameters is done by decision tree-based context clustering. Thereby, the context-dependent HMMs are assigned to different clusters depending on the combination of contexts using binary decision trees. Each cluster is characterized by a specific set of speech parameters. In this way, it is possible to estimate all HMM in a robust way with a typically sized training database. However, decision trees soon reach their limits when handling complex contexts. Only by increasing the size and in this way decreasing the efficiency of the decision tree, more complicated contexts (*e.g.* XOR) can be dealt with. In addition to that, decision trees require partitioned input data with each partition having a different set of parameters. In this way, with less data per region overfitting is likely to happen, which then results in a lack of quality. These downsides can be avoided by using a DNN instead of multiple decision trees. Nevertheless, this also introduces two disadvantages: The first one arises in terms of computational power. Both, in the training and the prediction stage, decision trees require much fewer operations (total amount and level of complexity) than DNN. The second one has to do with the decision process in its basic form. With decision trees, a binary question has to be answered, whilst a DNN consists of weighted neurons, which use non-linear activation functions (*e.g.* sigmoid, tanh, ReLU [8]) to determine their state. As a consequence, interpretable rules are far easier to produce with decision trees than with DNN. In Figure 2, the structure of a DNN-based speech synthesis system is shown. First, a sequence of input features is generated after analyzing the input text. These parameters contain numeric values like the number of words in a sentence or the duration of a phoneme as well as binary answers to questions like "Is the current phoneme *aa*?". Then, this parameter sequence is fed into the DNN where a mapping to output features is deployed by using forward propagation. The DNN has to be trained before with pairs of input and output features from a database. In the following steps, the speech parameters are extracted from the statistics of the output features, and the voice waveform in turn generated from the speech parameters. This is done in the same way as in the HMM-based system. For this system, the function blocks of text analysis, parameter generation, and waveform synthesis can be reused from an HMM-based system. Only the mapping from input features (*e.g.* linguistic contexts) to output features (spectral and excitation parameters) is implemented differently.

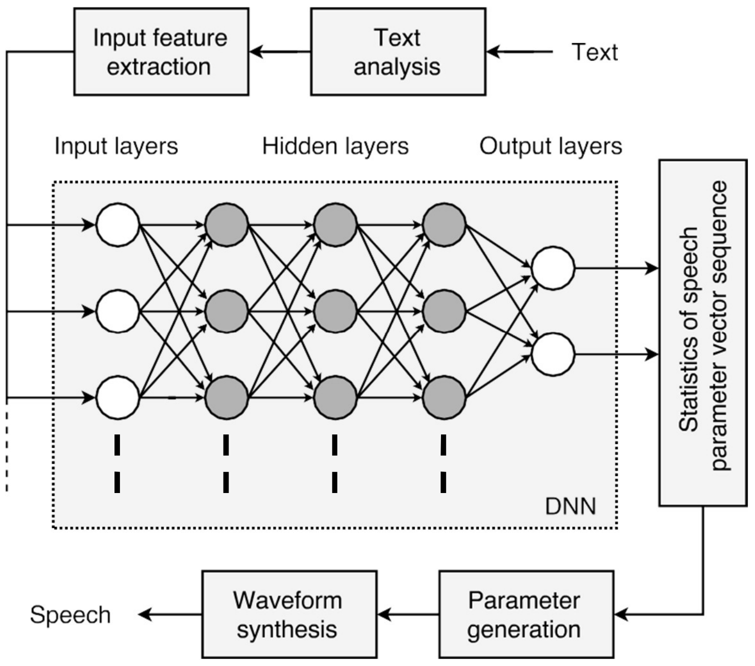

**Fig. 2.** DNN-based speech synthesis structure [17].

To compare the outcomes of the aforementioned framework with those of an HMM-based system, Zen *et al.* conducted experiments in their work on deep statistical speech synthesis [17]. They utilized a shared dataset comprising approximately 33,000 utterances in US English for both systems. The HMM-based system employed 2,554 questions for decision tree-based context clustering. To influence the size of the decision trees, a scaling factor $\alpha$ was employed, with higher values resulting in smaller decision trees. A value of $\alpha = 1$ denoted a typical HMM-based system. The input features of the DNN-based system encompassed 342 binary features (e.g., phoneme identities) and 25 numerical features (e.g., syllable count). The authors employed the sigmoid function as the activation function based on its superior performance observed in previous tests. Overall, multiple networks were utilized, each varying in the number of layers (1, 2, 3, 4, or 5) and the number of units per layer (256, 1024, or 2048).

The objective evaluation demonstrated that the DNN-based system outperformed the HMM-based system in voiced/unvoiced classification and aperiodicity prediction, regardless of the layer count or the units per layer. Regarding Mel-cepstral distortion, only DNN with three or more layers surpassed the HMM-based system. For the subjective evaluation, 173 test sentences were synthesized using both the DNN-based and the HMM-based systems. Listeners were then asked to choose their preferred system, with the option of selecting "neutral" if no difference was perceived. Both systems employed an equal number of parameters in this test. The DNN-based approach utilized four layers with varying numbers of units per layer. The subjective evaluation results indicated a clear preference for the speech samples generated by the DNN-based systems, irrespective of the units per layer. Listeners described them as having a less muffled quality.

In conclusion, it can be stated that a DNN-based approach to implementing the acoustic model of a speech synthesis system is a reasonable alternative to the conventional decision tree-based strategy. The improved performance for predicting spectral and excitation parameters and the more natural-sounding voice both show the potential of the DNN-based approach for speech synthesis. However, the higher computational costs both at the training and prediction stage due to more complex arithmetic operations in the DNN-based approach indicate where future work should be focused.

## 3.2 Other Ways for Improvement

The previous section focused on the impact a DNN has, by representing the acoustic model of a speech synthesis system. However, deep learning models can be deployed in other parts of SPSS as well. In [11], different ways to implement a DNN into a speech synthesis system are investigated. Therefore, different parts of an HMM-based speech synthesis system are once modeled with a DNN and once with the conventional technique. Then, the results are compared in an objective and a subjective fashion.

Hashimoto *et al.* concentrated on two core components of a speech synthesis system, the acoustic models and the speech generation part. In Figure 3, you can see the simplified structure of a speech synthesis system. In the first block, the text is analyzed and the contextual features are extracted (A). These are then converted to static and dynamic acoustic features (B) by the acoustic models. The parameter generation block then uses these acoustic features to create speech parameters (C). In the last step, the speech waveform is synthesized.

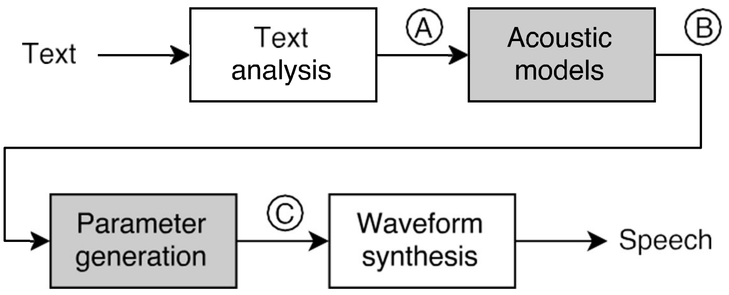

**Fig. 3.** Simplified structure of speech synthesis system [11].

The two gray-colored blocks in Figure 3 are the subject of two experiments conducted in [11]. The conventional approach for the acoustic models is the use of decision tree clustered HMM, whereas the parameter generation usually is implemented by an MLPG algorithm. As seen in Section 3.1, the acoustic models can be represented by a DNN. Hashimoto *et al.* also included this approach in their experiments but furthermore used a DNN for the speech parameter generation task. In Table 2, the systems resulting from the different combinations of the deployment of either conventional or DNN-based approaches are shown. Systems I - IV are part of Experiment 1 and all systems except System IV are part of Experiment 2. In experiment 1, all DNNs are equipped with three hidden layers and different numbers of units per layer (256, 512, or 1024), while the size of the decision trees was controlled similarly as in [17].

**Table 2.** Different systems within the experiment [11].

| **System Nr.** | **I** | **II** | **III** | **IV** | **V** | **VI** |
|---|---|---|---|---|---|---|
| Experiments | 1&2 | 1&2 | 1&2 | 1 | 2 | 2 |
| Acoustic models | HMM | HMM | DNN | DNN | HMM | HMM |
| Speech generation | MLGP | DNN | MLGP | DNN | DNN+MLGP | MLGP+DNN |

Only the Mel-cepstral distortion was used as objective evaluation. Here, the systems with a DNN (system II-IV) all outperformed system I in a similar way, as soon as they had 512 or more units per layer. For the subjective evaluation, 20 sentences were played back to eight listeners who had to assign a naturalness score between 1 and 5 (with 5 being the most natural). After applying the mean opinion score test method, system III was most preferred with 3.53 followed by system IV with

3.17 and system I with 3.08. The less natural-sounding voice according to the listeners was produced by system II. From these results, it can be concluded that the speech generation part implemented with MLPG (systems I & III) produces a more natural output than those with a DNN (systems II & IV).

Due to these outcomes, experiment 2 introduces two new test cases, system V and system VI. For the parameter generation task, a combination of MLPG and DNN is used, for which the respective input and output features were adapted. Again, an objective and a subjective evaluation like in experiment 1 were conducted to measure the performance. The objective evaluation showed the same trend as in experiment 1, as soon as 512 or more units per layer were used, the Mel-cepstral distortion was smaller than with the conventional approach (system I). In the subjective evaluation, the two new systems (V & VI) both were preferred over system I, with system VI being the most preferred system of this experiment. System III again was better and system II again was evaluated less natural than system I.

In conclusion, we see that the experiments conducted in [11] confirmed the results of [17] that the decision tree-clustered HMM can be replaced by a DNN to improve both the prediction performance as well as the naturalness of the generated speech. Beyond that, Hashimoto *et al.* demonstrated that it is worthwhile to go one step further and deploy an additional DNN as a post-filtering block into the parameter generation task of the speech synthesis system to further improve the above-stated results.

## 4 Speech Synthesis on Mobile Devices

In this section, two approaches to optimize speech synthesis for mobile devices are presented. In the first, some optimization steps on a conventional HMM-based system are suggested [15], whereas the second approach introduces deep learning models to get rid of the dependence on Internet-based services [6].

### 4.1 Motivation and Challenges

During the last 10 years, advances in technology have led to a tremendous spread of mobile devices, especially smartphones. According to [2], there were 2.1 billion smartphone users worldwide in 2016 which is forecasted to increase to almost 3 billion in 2020. In [2], it also states that only in 2016 about 1.5 billion new smartphones were sold. From these numbers, we can derive that already today and with increasing significance in the near future, smartphones constitute an essential part of our daily life. Especially on smartphones where the visual output is restricted to a rather small screen, usually not bigger than 5 inches in diagonal, speech-enabled interaction can improve the user experience in a significant way. In [15], three application scenarios of how speech interaction can assist are pointed out. Firstly, speech interaction can be used as an extension to existing communication channels. Secondly, if other outputs are restricted (*e.g.* while driving), speech can be deployed as the main output. Thirdly, speech can help visually impaired or blind people to interact with a system at all (*e.g.* screen readers).

However, when implementing a speech synthesis application on a mobile device, the increased computational power and storage capacity on recent smartphones cannot be used as with desktop devices. One aspect is the power consumption. An application on a smartphone should use as little processing power as possible to avoid shortening the battery life unnecessarily [15]. Another issue is the restricted amount of main memory (16 - 32 MB RAM per app), which forces each application to be as resource-saving as possible. While accomplishing these demands, the output speech still needs to be processed in real time to ensure a good user experience [6].

## 4.2 Optimized HMM-based Synthesis

One approach to optimizing HMM-based speech synthesis for mobile devices can be seen in [15]. Here, Tóth *et al.* conduct a study to optimize an HMM-based speech synthesis system in terms of computational power, playback latency, and footprint size while ensuring a reasonable quality of the synthesized speech. In the following, this approach is explained further.

As already pointed out in Section 2.1, HMM-based speech synthesis offers the best trade-off between speech quality and footprint size. This surely makes it the most appropriate approach to be deployed in mobile systems. Tóth *et al.* indicate that there already have been several approaches to optimize HMM-based speech synthesis. However, they claim to be the first who emphasize the optimization steps on resource-constrained platforms to achieve a reduction in computational costs, while still fulfilling real-time responsiveness. To measure the processing time of the speech synthesis procedure, Tóth *et al.* divide the TTS process into three steps: The loading of the HMM database into main memory (1), the speech parameter generation (2), and the waveform synthesis (3). A 17-second sequence is chosen as a test sentence, although it is unlikely that such a long utterance has to be synthesized often in real life (since in practice segments with one to five seconds are synthesized). That way, it can be ensured that shorter sequences are synthesized as desired. The following four approaches to optimize HMM-based TTS are suggested by Tóth *et al.*: **Adjusting vocoder parameters, Reducing the size of decision trees, Introducing streaming synthesis, and Applying source code optimizations.**

In the first approach, the conventional MLSA filter used for speech parameter generation is replaced by an LSP filter. In this way, the performance in step (2) can be enhanced. Different orders of the LSP filter are tested to compare speech subjective quality, with an order of 18 being the conventional setup and 14, 12, and 10 being the optimization approaches. The second approach dealt with the size of the decision trees. While lowering the number of nodes of a decision tree reduces the quality of synthesized speech, it also decreases the memory footprint and the computational costs. Therefore, three different test settings were defined. Table 3 displays the three different test settings.

**Table 3.** Test settings of optimization approach [15].

| Setting | Nodes in decision trees | Footprint size |
|---|---|---|
| Original | 6983 | 666 KB |
| Setting 1 | 4762 | 463 KB |
| Setting 2 | 2743 | 214 KB |
| Setting 3 | 1273 | 140 KB |

General speech synthesis frameworks usually do not include the audio playback functionality to ensure platform independence. Implementing this part allows streaming synthesis. In this way, one already synthesized segment is played back while the next segment is generated. While long segments cause an undesired latency, too short segments result in incontiguous audio playback. The optimal segment size is determined at run-time to achieve a compromise between low as possible latency and low as possible computational effort. Low-resource devices differ from high-end devices not only in the amount of available memory and computing performance but also in the chip architecture. Memory management, fixed- versus floating-point calculation, and conditional call management are points to be aware of. By implementing these issues in an architecture-optimized way, the processing load and time can be decreased. The above-described optimizations were tested on three different smartphones with increasing CPU speed and main memory, starting with the "weakest": an Apple iPhone 3G, a Samsung Galaxy Spica (GT-i5700), and a HTC Desire (A8181). Therefore, the above-described optimization steps were tested one by one, since they do not affect each other in a noteworthy manner.

For the first approach, a 12th-order LSP filter was chosen as the optimal setting in terms of latency and subjective speech quality. Concerning the size of the decision trees, setting 1 proved to be the optimal setting, since the time to load the HMM database (1) was decreased to half as before without notable impaired speech quality. The streaming synthesis approach tremendously decreased the duration of step (1), with an improvement from the order of seconds to 8 ms without affecting the speech quality at all. Concerning the source code optimizations, almost no optimizations have been conducted. Only some conditional calls were removed, which resulted in an insignificant improvement in latency. In total, the computation time could be decreased by 65 % while the speech quality was impaired only negligibly.

## 4.3 Deep Learning-based Synthesis

The usual approach for current VPA (or generally HCI interfaces) to use servers for processing input data and generating dedicated output actions, results in a total dependency on existing Internet access. In [6], Boros *et al.* indicate three major disadvantages of this strategy: The availability of network access is not always granted (*e.g.* bad coverage, additional charges abroad), the issue of data security, and possible delays due to bad network connection causing bad user experience. The authors are therefore suggesting the use of deep learning models to port an existing TTS system (based on Romanian language) to a mobile device. Therefore, the main goal is to reduce the total memory footprint without losing too much accuracy compared to the original system implementation. A TTS system consists of a text processing front-end and a digital signal-processing back-end. The suggestions made in [6] solely focus on the front end. In this scope, a deep learning model is introduced in each of the following text-processing tasks: **Syllabification (SYL), Phonetic transcription (PT), Part-of-speech tagging (POT), and Lexical stress prediction (LSP).**

All experiments are conducted with a DNN with two or three hidden layers respectively. Each hidden layer consists of an autoencoder. In the following, the experiments are explained further. Table 4 summarizes the results obtained when introducing a deep learning model to various text-processing tasks. As a reference system, the MIRA is used [7].

In the first task, the syllabification, each word is decomposed into syllables (phonological units). This contributes to the later conducted prosody generation. By using a data-driven approach instead of a rule-based approach, language independence is ensured. In comparison to the MIRA algorithm, the word-level accuracy of the DNN-based approach (two hidden layers) is only 0.78 % worse but enables a reduction of the model size by a factor of more than 250 to 36.7 KB. The second task, the phonetic transcription, is the process of translating letters to phonemes (also referred to as letter-to-sound or grapheme-to-phoneme) and is also chosen to be implemented in a data-driven approach, due to the same reason as in the syllabification. By using a DNN with two hidden layers, a word-level accuracy of 96.16 % can be achieved, which is a decrease of only 0.13 % compared to the MIRA-based technique. However, the model size again is reduced significantly from around 1.4 MB to 43.4 KB. To determine how a word is spoken, it's part-of-speech tagging (the third task) is one of the basic features. One important aspect of allocating a uniquely interpretable tag to each word in a sentence is to identify the correct pronunciation of a homograph. In this part, two DNNs were deployed (both with two hidden layers), one to replace the lexicon containing the mapping of words to their syntactic information and one that performs the tagging instead of a standard feed-forward NN. In this task, the biggest reduction of the model size could be achieved, from almost 100 MB to about 178 KB. Thereby, the word-level accuracy was reduced by 3.03 %. This reduction is due to the lack of fine-tuning of the DNN, which in this scope has not been conducted. The last task is lexical stress prediction, which represents an essential part of prosody generation, by indicating which syllables have to be given a special emphasis during the utterance. The DNN used for this task is equipped with three hidden layers and results in an overall accuracy of 97.67 % and a reduction of the model size by a factor of 60 to around 110 KB.

**Table 4.** Resulting accuracy and footprint size [6].

| Task | **SYL** | | **PT** | | **POT** | | **LSP** | |
|---|---|---|---|---|---|---|---|---|
| | MIRA | DNN | MIRA | DNN | NN | DNN | MIRA | DNN |
| Accuracy | 99.01% | 98.23% | 96.29% | 96.16% | 98.19% | 95.16% | 98.80 % | 97.67% |
| Size | 9.4 MB | 36.7 KB | 1.4 MB | 43.4 KB | 96 MB | 178 KB | 6 MB | 110 KB |

Overall, it can be concluded that the measures introduced in [6] enabled a tremendous reduction of the model sizes while keeping the performance almost as high as in the previous models.

## 5 Discussion

In relation to optimizing speech synthesis for resource-constrained platforms like mobile devices, we discussed two approaches in section 4. The second approach, involving deep learning, as discussed in section 4.3, is certainly the more promising of the two. The primary objective when considering mobile systems is to reduce the overall memory footprint, making it more practical for everyday use. Based on the results presented in Table 4, we can conclude that the integration of deep learning models into the front-end or text-processing components of a TTS system successfully achieves this goal without the need for internet access. Therefore, deep learning presents a highly reasonable strategy for adapting speech synthesis systems to mobile devices.

## 6 Conclusions

In this paper, a systematic review is presented focusing on the impact of deep learning on speech synthesis for mobile devices. In this context, first, different approaches to improve the conventional HMM-based synthesis have been pointed out, including the deployment of DNN as acoustic models or in the parameter generation task. These approaches have led to a significant improvement in the prediction performance and the speech quality [17, 11]. The study examines the implementation of speech synthesis algorithms in resource-constrained environments, such as mobile devices, and outlines two strategies: one with and one without the use of deep learning models. In the first one, the focus is set on adjusting different parameters and applying several optimization steps [15] in order to achieve real-time playback. As a result, the computation time can be decreased by 65 % without a negligible loss of speech quality. The second approach [6] tackles the dependency of network access while using speech synthesis applications on mobile devices. Therefore, the authors have suggested the use of DNN in several parts of the front-end of a TTS system so as to reduce the footprint size which enables the offline use of speech synthesis applications.

It is expected that the acceptance of VPA like Apple's Siri or Amazon's Alexa will be widespread in the near future [4]. Hence, the development of robust and resource-efficient speech synthesis methods is an essential part of meeting the challenges of mobile environments. Innovative technologies like personalized speech-to-speech translation or voice cloning are only two examples of emerging techniques, wherefore speech synthesis has to be as evolved as possible [16]. The utilization of deep learning methods towards the potential to accelerate future advancements in the field of speech synthesis for mobile devices. Consequently, it is worthwhile to further improve these methods both in terms of voice quality and efficient embedded implementation.

## References

1. The benefits of text-to-speech, https://www.readspeaker.com/blog/benefits-of-text-to-speech/, last accessed on 2023/05/22
2. Number of smartphone users worldwide, https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/, last accessed on 2023/06/02
3. Stephen Hawking - the computer, http://www.hawking.org.uk/%20the-computer.html, last accessed on 2023/06/11
4. Survey on usage of virtual personal assistants by Gartner http://www. gartner.com/newsroom/id/3551217, last accessed on 2023/06/18

5. A. W. Black, H. Zen and K. Tokuda, "Statistical Parametric Speech Synthesis," 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07, Honolulu, HI, USA, 2007, pp. IV-1229-IV-1232, doi: 10.1109/ICASSP.2007.367298. (April 2007).
6. Tiberiu Boroş and Stefan Daniel Dumitrescu. 2015. Robust deep-learning models for text-to-speech synthesis support on embedded devices. In Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems (MEDES '15). Association for Computing Machinery, New York, NY, USA, 98–102. https://doi.org/10.1145/2857218.2857234 (October 2015).
7. Boros, T.: A unified lexical processing framework based on the margin infused relaxed algorithm. A case study on the romanian language. In: Recent Advances in Natural Language Processing, RANLP 2013, 9-11 September, 2013, Hissar, Bulgaria. pp. 91–97 (September 2013)
8. Hoon Chung, Sung Joo Lee and Jeon Gue Park, "Deep neural network using trainable activation functions," 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, 2016, pp. 348-352,
9. doi: 10.1109/IJCNN.2016.7727219. (July 2016)
10. Li Deng (2014), "A tutorial survey of architectures, algorithms, and applications for deep learning", APSIPA Transactions on Signal and Information Processing: Vol. 3: No.1, e2. http://dx.doi.org/10.1017/atsip.2013.9-(Jan 2014)
11. Dhavala, L., 2014. Use of synthetic voice to improve communication between air traffic controllers and pilots. In AVIATION MANAGEMENT CONFERENCE e.2, (p. 28). ( November-2014)
12. K. Hashimoto, K. Oura, Y. Nankaku and K. Tokuda, "The effect of neural networks in statistical parametric speech synthesis," *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, South Brisbane, QLD, Australia, 2015, pp. 4455-4459, doi: 10.1109/ICASSP.2015.7178813. (April 2015)
13. Hinterleitner, F.: Quality of Synthetic Speech: Perceptual Dimensions, Influencing Factors, and Instrumental Assessment (T-Labs Series in Telecommunication Services). Springer, Singapore, ed.1‚ 2017 (April, 2017).
14. Z. -H. Ling *et al.*, "Deep Learning for Acoustic Modeling in Parametric Speech Generation: A systematic review of existing techniques and future trends," in *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 35-52, May 2015, doi: 10.1109/MSP.2014.2359987.(May, 2015)
15. Suendermann, D., Höge, H., Black, A. (2010). Challenges in Speech Synthesis. In: Chen, F., Jokinen, K. (eds) Speech Technology. Springer, New York, NY. https://doi.org/10.1007/978-0-387-73819-2_2 (2010)
16. Tóth, B. and Németh, G., 2012. Optimizing HMM speech synthesis for low-resource devices. Journal of Advanced Computational Intelligence Vol, 16(2).( November 2011)
17. Yamagishi, J.: New and emerging applications of speech synthesis, https://www.pure.ed.ac.uk/ws/portalfiles/portal/117247223/Simon_King_Scale2012_Tutorial_refined_for_publication.pdf, last accessed on 2023/08/16
18. H. Zen, A. Senior and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 2013, pp. 7962-7966, doi: 10.1109/ICASSP.2013.6639215.(May,2013)