What Concept-to-Speech Can Gain for Prosody

Markus Schnell
Laboratory for Emerging Technologies
Infineon Technologies AG, Munich
markus.schnell@infineon.com

Rüdiger Hoffmann
Laboratory of Acoustics and Speech Communication
Dresden University of Technology
ruediger.hoffmann@ias.et.tu-dresden.de

Abstract
This article proposes a concept-to-speech system with automated prosody learning based on reinforcement learning. The concept-to-speech system, named Demosthenes, is an extension of the text-to-speech system DreSS. Demosthenes is responsible for template-based text generation and symbolic prosody prediction, while DreSS takes care of acoustic prosody and speech synthesis. The prosody predictor is an application of reinforcement learning, using content, given and new, contrast, and number of words since last accented words as indicators in state space. The system is trained with a simple rule, giving reward according to prediction performance on a small sample text. For an impression of the gain in prosodic quality, we compare the concept-to-speech system to an existing text-to-speech system. The results indicate a clear preference for the concept-to-speech system.

1. Introduction
It is widely believed that prosody in speech synthesis should benefit from concept-to-speech generation. This assumption is based on two reasons: First, there is no need to analyze a text, say, for part-of-speech or pronunciation, as this knowledge is already present in the generation component. Second, concept-to-speech enables access to information like contrast, which cannot be detected in raw text without sophisticated world knowledge.

There has been some activity in constructing concept-to-speech systems, beginning in 1979 with the work of Young and Fallside [1]. They use syntactic structure to derive boundaries and frequency values. Later works introduce a symbolic interface to prosody (accents, tones) and utilize discourse, information or argument structure for prosody prediction [2, 3, 4, 5].

However, all these systems share a common flaw: they rely on carefully crafted, hand-written rules by linguistic experts. This is not necessarily a disadvantage for building a working system, but it places too much effort on human coding, thereby increasing cost for system development. This is one reason why currently most concept-to-speech systems exist for English only, and some for Dutch and German.

This is a variant of the scaling problem, in the words of Richard Sutton [6]:

"An AI system too reliant on manual tuning, for example, will not be able to scale past what can be held in the heads of a few programmers."

Using a learning system instead of hand-written rules would make adaption to other languages easier and allow different prosodic strategies on different occasions.

In fact, Shimei Pan has made significant progress, first in using corpus-based statistics for generating rules [7], later in using a corpus directly for instance-based prosody [8]. Still, these systems rely on annotated corpora, often created for just this purpose, which is a time-consuming and costly process.

This paper wants to explore another approach: An existing text-to-speech system [9] is extended with a generation engine, which includes a module for prosody prediction. This module uses reinforcement learning to establish a prosody strategy. In the following, we explain our approach for the example of (pitch) accenting.

2. The production system
The generation system Demosthenes\(^1\) creates spoken monologues, describing and comparing different kinds of vine grape and their respective wines (Riesling, Pinot noir, Zinfandel, ...). The target language is German, but the prosody module can be trained in the same way as described below for other languages. The modeled prosody features include sentence mood, phrase boundaries and accent placement.

Demosthenes is derived from an existing text-to-speech system, namely DreSS [9], which is depicted in Fig. 1. DreSS is organized in different levels from raw text to speech signal. Each level is accessible from the outside with a special format language.

\(^1\)Demosthenes (384–322 BC), one of the greatest ancient orators, is said to have overcome his speech impediment by talking with a mouth full of stones and trying to drown out the brawl of the sea.
To convert this architecture into a concept-to-speech system, two modules are completely removed: the text analyzer and the letter-to-sound converter. The information from these modules is now generated by the planner/formulator. The symbolic prosody predictor is changed to work with the new architecture. The acoustic prosody predictor and the synthesis engine are executed as in the text-to-speech system. The architecture for Demosthenes is given in Fig. 2.

The planner is responsible for selecting and structuring content from a database, depending on the chosen task. Instead of using a grammar system, Demosthenes employs ideas from template based generation systems [10, 11]. The planner has access to sentence templates like

\begin{verbatim}
wein ist modifier eine sorte  (Template)
\end{verbatim}

Each template contains information about topic, inflection, sentence mood, punctuation and boundaries, which are not shown here. The formulator converts these templates into a format called utterance score. It contains all information needed for generating the final output, which is the responsibility of the articulator.

3. Reinforcement learning

The symbolic prosody predictor transforms indicators to mark-up codes with reinforcement learning, an algorithm inspired from psychology [12]. A reinforcement learning system consists of an agent, an environment and state, action and reward information, as shown Fig. 3. Instead of directly telling the agent what it should do at each step, it tries by itself and evaluates its strategy based on the reward information the environment gives after each action. The agent tries to maximize the cumulative reward over time.

The agent maintains a table with estimates for the cumulative reward for each state and every possible action in that state. At each time step, the agent observes the current state \( s \) and chooses the action \( a \) which has the biggest value in the table for this state (exploitation). Every once in a while, however, the agent takes another action by random selection (exploration).

The action is executed and results in a reward \( r \) and a new state \( s' \) from the environment. The agent can now update its state-action-table \( Q \) for state \( s \) and action \( a \) with

\[ \Delta Q(s, a) = \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \]  \hspace{1cm} (1)

The learning parameters \( \alpha \) and \( \gamma \) are taken from the interval [0,1].

4. Indicator generation

The state table consists of several indicators. To avoid a huge state space with high dimensionality, we consider only a few of the many indicators described in the literature for prosody.
prediction [13, 14, 15, 16]. The chosen indicators are content, given and new, contrast, and number of words since last accented word.

4.1. Content

The distinction of content and function words has long been known in speech synthesis, since it is easy to infer. That fact is related to an analysis by Pan. She showed that word informativeness is correlated with accentability [7]. Content and function word are the two halves of a scale of word informativeness. Examples for function words are *ist, der* and *oder*, examples for content words are *Pfirsich, Weißweinsorte* and *duftet*. The content state of a word is part of its dictionary information.

4.2. Given and new

The distinction between given and new is founded on the observation that already introduced discourse objects tend to be deaccented, while newly introduced objects tend to be accented. Demosthenes keeps track of the mentioned objects and marks the first occurrence as new, and all following as given. This marking is independent of the chosen lexicalisation for the object.

4.3. Contrast

Sometimes already mentioned words are accented, in particular when they are used in contrastive statements [3, 17]. Demosthenes marks a word as contrastive if it is in contrast to an already existing object.

4.4. Number of words since last accented word

Speakers seem to avoid long sequences of unaccented words. This observation is the result of an analysis of a richly annotated corpus of German radio news [18, 19]. Fig. 4 shows how often sequences of different length occur. In the examined corpus there are never more than nine unaccented words in sequence. Most accents tend to be placed within the next three words of an accented word. Demosthenes takes this into account by keeping a counter of the number of words since the last accented word. The counter is set to zero whenever a word gets accented.

5. Training

Normally, reward would be given according to the quality of the speech signal. However, to speed up the learning process, we introduced a dedicated training phase. Demosthenes generated a short text with 75 words. This text was read by a native speaker of German and recorded for analysis. Each word was either marked as accented or unaccented. The resulting accentuation was transferred into the learning system, and used together with the following reward rule:

\[
\text{(Reward Rule)}
\]

```plaintext
IF current text is same as recorded
  THEN
    IF current word is accented or unaccented
      in correspondence with recording
      THEN give a reward of 1.0
      ELSE give a reward of -1.0
    ELSE give a reward of 0.0

IF no recorded text exists for the current text, a reward of 0.0 is given. Otherwise, a reward of 1.0 is given if the accent prediction was correct, and a reward of -1.0 if it was not correct.
```

6. Comparing text-to-speech with concept-to-speech

Is this approach an improvement against standard speech synthesis? To answer this question, we conducted an experiment. A text with specific challenges was selected to highlight the differences between text-to-speech and concept-to-speech.

The basic algorithm employed in the text-to-speech system DreSS for accent prediction is this: If the word in question is a content word, accent it. Content word or not is derived from a comprehensive dictionary. To smooth accenting, there is an additional rule, considering the distance to the last accent.

Demosthenes synthesized five different texts with the proposed algorithm. The same texts were synthesized by the text-to-speech system DreSS. The texts had between 35 and 79 words. Each text pair was presented to a number of native speakers of German in random order. For each text, the hearer had to indicate which version of a pair he preferred. Table 1 shows the results.

The preference is clearly for concept-to-speech. Concluding from the comments the participants made, what they didn’t like about the text-to-speech system was the “strange placement of accents” and “no boundaries”. On the other hand, one participant claimed one concept-to-speech exam-
Table 1: Preference for text-to-speech or concept-to-speech.

<table>
<thead>
<tr>
<th>Example</th>
<th>text-to-speech</th>
<th>concept-to-speech</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

ple as “too pronounced”.

7. Conclusions

This paper described the application of reinforcement learning to prosody prediction for a concept-to-speech system. In contrast to earlier systems, the accenting rules were learnt automatically. Only a short text was needed for training. An experiment showed a clear improvement against text-to-speech. We conclude that our approach makes it possible to have pleasant prosody without the need for prosody experts.

8. References


