Statistical Modeling for Unit Selection in Speech Synthesis

Cyril Allauzen and **Mehryar Mohri** and **Michael Riley***
AT&T Labs – Research
180 Park Avenue, Florham Park, NJ 07932, USA
{allauzen, mohri, riley}@research.att.com

Abstract

Traditional concatenative speech synthesis systems use a number of heuristics to define the target and concatenation costs, essential for the design of the unit selection component. In contrast to these approaches, we introduce a general statistical modeling framework for unit selection inspired by automatic speech recognition. Given appropriate data, techniques based on that framework can result in a more accurate unit selection, thereby improving the general quality of a speech synthesizer. They can also lead to a more modular and a substantially more efficient system.

We present a new unit selection system based on statistical modeling. To overcome the original absence of data, we use an existing high-quality unit selection system to generate a corpus of unit sequences. We show that the concatenation cost can be accurately estimated from this corpus using a statistical n-gram language model over units. We used weighted automata and transducers for the representation of the components of the system and designed a new and more efficient composition algorithm making use of *string potentials* for their combination. The resulting statistical unit selection is shown to be about 2.6 times faster than the last release of the AT&T Natural Voices Product while preserving the same quality, and offers much flexibility for the use and integration of new and more complex components.

1 Motivation

A concatenative speech synthesis system (Hunt and Black, 1996; Beutnagel et al., 1999a) consists of three components. The first component, the *text-analysis frontend*, takes text as input and outputs a sequence of feature vectors that characterize the acoustic signal to synthesize. The first element of each of these vectors is the predicted phone or half-phone; other elements are features such as the phonetic context, acoustic features (e.g., pitch, duration), or prosodic features.

The second component, *unit selection*, determines in a set of recorded acoustic units corresponding to phones (Hunt and Black, 1996) or half-phones (Beutnagel et al., 1999a) the sequence of units that is the *closest* to the sequence of feature vectors predicted by the text analysis frontend. The final component produces an acoustic signal from the unit sequence chosen by unit selection using simple concatenation or other methods such as PSOLA (Moulines and Charpentier, 1990) and HNM (Stylianou et al., 1997).

Unit selection is performed by defining two cost functions: the *target cost* that estimates how the features of a recorded unit match the specified feature vector and the *concatenation cost* that estimates how well two units will be perceived to match when appended. Unit selection then consists of finding, given a specified sequence of feature vectors, the unit sequence that minimizes the sum of these two costs.

The target and concatenation cost functions have traditionally been formed from a variety of heuristic or *ad hoc* quality measures based on features of the audio and text. In this paper, we follow a different approach: our goal is a system based purely on statistical modeling. The starting point is to assume that we have a training corpus of utterances labeled with the appropriate unit sequences. Specifically, for each training utterance, we assume available a sequence of feature vectors $f = f_1 \dots f_n$ and the corresponding units $u = u_1 \dots u_n$ that should be used to synthesize this utterance. We wish to estimate from this corpus two probability distributions, $P(f|u)$ and $P(u)$. Given these estimates, we can perform unit selection on a novel utterance using:

$$u = \operatorname*{argmax}_u P(u|f) \quad (1)$$

$$= \operatorname*{argmin}_u \left(-\log P(f|u) - \log P(u)\right) \quad (2)$$

Equation 1 states that the most likely unit sequence is selected given the probabilistic model used. Equation 2 follows from the definition of conditional probability and that $P(f)$ is fixed for a given utterance. The two terms appearing in Equation 2 can be viewed as the statistical counterparts

* This author's new address is: Google, Inc, 1440 Broadway, New York, NY 10018, riley@google.com.

of the target and concatenation costs in traditional unit selection.

The statistical framework just outlined is similar to the one used in speech recognition (Jelinek, 1976). We also use several techniques that have been very successfully applied to speech recognition. For instance, in this paper, we show how $-\log P(u)$ (the concatenation cost) can be accurately estimated using a statistical n-gram language model over units. Two questions naturally arise.

(a) How can we collect a training corpus for building a statistical model? Ideally, the training corpus could be human-labeled, as in speech recognition and other natural language processing tasks. But this seemed impractical given the size of the unit inventory, the number of utterances needed for good statistical estimates, and our limited resources. Instead, we chose to use a training corpus generated by an existing high-quality unit selection system, that of the AT&T Natural Voices Product. Of course, building a statistical model on that output can, at best, only match the quality of the original. But, it can serve as an exploratory trial to measure the quality of our statistical modeling. As we will see, it can also result in a synthesis system that is significantly faster and modular than the original since there are well-established algorithms for representing and optimizing statistical models of the type we will employ. To further simplify the problem, we will use the existing traditional target costs, providing statistical estimates only of the concatenation costs ($-\log P(u)$).

(b) What are the benefits of a statistical modeling approach?

(1) **High-quality cost functions**. One issue with traditional unit selection systems is that their cost functions are the result of the following compromise: they need to be complex enough to have a perceptual meaning but simple enough to be computed efficiently. With our statistical modeling approach, the labeling phase could be performed offline by a highly accurate unit selection system, potentially slow and complex, while the run-time statistical system could still be fast. Moreover, if we had audio available for our training corpus, we could exploit that in the initial labeling phase for the design of the unit selection system.

(2) **Weighted finite-state transducer representation**. In addition to the already mentioned synthesis speed and the opportunity of high-quality measures in the initial offline labeling phase, another benefit of this approach is that it leads to a natural representation by weighted transducers, and hence enables us to build a unit selection system using general and flexible representations and methods already in use for speech recognition, e.g., those found in the FSM (Mohri et al., 2000), GRM (Allauzen et al., 2004) and DCD (Allauzen et al., 2003) libraries. Other unit selection systems based on weighted transducers were also proposed in (Yi et al., 2000; Bulyko and Ostendorf, 2001).

(3) **Unit selection algorithms and speed-up**. We present a new unit selection system based on statistical modeling. We used weighted automata and transducers for the representation of the components of the system and designed a new and efficient composition algorithm making use of *string potentials* for their combination. The resulting statistical unit selection is shown to be about 2.6 times faster than the last release of the AT&T Natural Voices Product while preserving the same quality, and offers much flexibility for the use and integration of new and more complex components.

2 Unit Selection Methods

2.1 Overview of a Traditional Unit Selection System

This section describes in detail the cost functions used in the AT&T Natural Voices Product that we will use as the baseline in our experimental results, see (Beutnagel et al., 1999a) for more details about this system. In this system, unit selection is based on (Hunt and Black, 1996) but using units corresponding to halfphones instead of phones. Let U be the set of recorded units. Two cost functions are defined: the *target cost* $C_t(f_i, u_i)$ is used to estimate the mismatch between the features of the feature vector f_i and the unit u_i; the *concatenation cost* $C_c(u_i, u_j)$ is used to estimate the smoothness of the acoustic signal when concatenating the units u_i and u_j. Given a sequence $f = f_1 \dots f_n$ of feature vectors, unit selection can then be formulated as the problem of finding the sequence of units $u = u_1 \dots u_n$ that minimizes these two costs:

$$u = \underset{u \in U^n}{\operatorname{argmin}} \left(\sum_{i=1}^{n} C_t(f_i, u_i) + \sum_{i=2}^{n} C_c(u_{i-1}, u_i) \right)$$

In practice, not all unit sequences of a given length are considered. A preselection method such as the one proposed by (Conkie et al., 2000) is used. The computation of the target cost can be split in two parts: the context cost C_p that is the component of the target cost corresponding to the phonetic context, and the feature cost C_f that corresponds the

other components of the target cost:

$$C_t(f_i, u_i) = C_p(f_i, u_i) + C_f(f_i, u_i) \qquad (3)$$

For each phonetic context ρ of length 5, a list $L(\rho)$ of the units that are the most frequently used in the phonetic context ρ is computed. For each feature vector f_i in f, the candidate units for f_i are computed in the following way. Let ρ_i be the 5-phone context of f_i in f. The context costs between f_i and all the units in the preselection list of the phonetic context ρ_i are computed and the M units with the best context cost are selected:

$$U_i = \underset{u_i \in L(\rho_i)}{\text{M-best}}(C_p(f_i, u_i))$$

The feature costs between f_i and the units in U_i are then computed and the N units with the best target cost are selected:

$$U'_i = \underset{u_i \in U_i}{\text{N-best}}(C_p(f_i, u_i) + C_f(f_i, u_i))$$

The unit sequence u verifying:

$$u = \underset{u \in U'_1 \cdots U'_n}{\operatorname{argmin}} \left(\sum_{i=1}^{n} C_t(f_i, u_i) + \sum_{i=2}^{n} C_c(u_{i-1}, u_i)\right)$$

is determined using a classical Viterbi search. Thus, for each position i, the N^2 concatenation costs between the units in U'_i and U'_{i+1} need to be computed. The caching method for concatenation costs proposed in (Beutnagel et al., 1999b) can be used to improve the efficiency of the system.

2.2 Statistical Modeling Approach

Our statistical modeling approach was described in Section 1. As already mentioned, our general approach would consists of deriving both the target cost $-\log P(f|u)$ and the concatenation cost $-\log P(u)$ from appropriate training data using general statistical methods. To simplify the problem, we will use the existing target cost provided by the traditional unit selection system and concentrate on the problem of estimating the concatenation cost.

We used the unit selection system presented in the previous section to generate a large corpus of more than 8M unit sequences, each unit corresponding to a unique recorded halfphone. This corpus was used to build an n-gram statistical language model using Katz backoff smoothing technique (Katz, 1987). This model provides us with a new cost function, the *grammar cost* C_g, defined by:

$$C_g(u_k|u_1 \ldots u_{k-1}) = -\log(P(u_k|u_1 \ldots u_{k-1}))$$

where P is the probability distribution estimated by our model. We used this new cost function to replace both the concatenation and context costs used in the traditional approach. Unit selection then consists of finding the unit sequence u such that:

$$u = \underset{u \in U^n}{\operatorname{argmin}} \sum_{i=1}^{n} (C_f(f_i, u_i) + C_g(u_i|u_{i-k} \ldots u_{i-1}))$$

In this approach, rather than using a preselection method such as that of (Conkie et al., 2000), we are using the statistical language model to restrict the candidate space (see Section 4.2).

3 Representation by Weighted Finite-State Transducers

An important advantage of the statistical framework we introduced for unit selection is that the resulting components can be naturally represented by weighted finite-state transducers. This casts unit selection into a familiar schema, that of a Viterbi decoder applied to a weighted transducer.

3.1 Weighted Finite-State Transducers

We give a brief introduction to weighted finite-state transducers. We refer the reader to (Mohri, 2004; Mohri et al., 2000) for an extensive presentation of these devices and will use the definitions and notation introduced by these authors.

A *weighted finite-state transducer* T is an 8-tuple $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ where Σ is the finite input alphabet of the transducer, Δ is the finite output alphabet, Q is a finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, $E \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Delta \cup \{\epsilon\}) \times \mathbb{R} \times Q$ a finite set of transitions, $\lambda : I \to \mathbb{R}$ the initial weight function, and $\rho : F \to \mathbb{R}$ the final weight function mapping F to $\mathbb{R}$. In our statistical framework, the weights can be interpreted as log-likelihoods, thus there are added along a path. Since we use the standard Viterbi approximation, the weight associated by T to a pair of strings $(x, y) \in \Sigma^* \times \Delta^*$ is given by:

$$[\![T]\!](x, y) = \min_{\pi \in R(I,x,y,F)} \lambda[p[\pi]] + w[\pi] + \rho[n[\pi]]$$

where $R(I, x, y, F)$ denotes the set of paths from an initial state $p \in I$ to a final state $q \in F$ with input label x and output label y, $w[\pi]$ the weight of the path π, $\lambda[p[\pi]]$ the initial weight of the origin state of π, and $\rho[n[\pi]]$ the final weight of its destination. A *Weighted automaton* $A = (\Sigma, Q, I, F, E, \lambda, \rho)$ is defined in a similar way by simply omitting the output (or input) labels. We denote by $\Pi_2(T)$ the

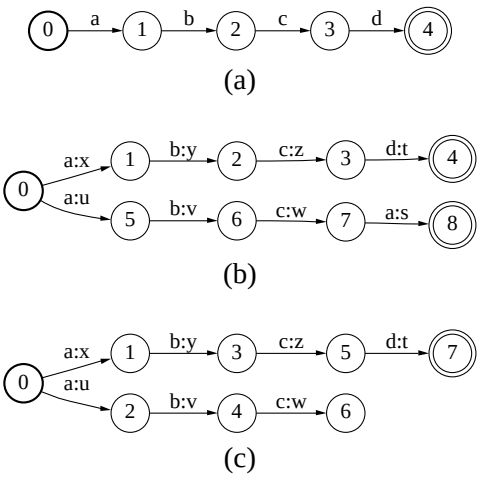

Figure 1: (a) Weighted automaton T_1. (b) Weighted transducer T_2. (c) $T_1 \circ T_2$, the result of the composition of T_1 and T_2.

weighted automaton obtained from T by removing its input labels.

A general *composition* operation similar to the composition of relations can be defined for weighted finite-state transducers (Eilenberg, 1974; Berstel, 1979; Salomaa and Soittola, 1978; Kuich and Salomaa, 1986). The composition of two transducers T_1 and T_2 is a weighted transducer denoted by $T_1 \circ T_2$ and defined by:

$$[\![T_1 \circ T_2]\!](x, y) = \min_{z \in \Delta^*} \{[\![T_1]\!](x, z) + [\![T_2]\!](z, y)\}$$

There exists a simple algorithm for constructing $T = T_1 \circ T_2$ from T_1 and T_2 (Pereira and Riley, 1997; Mohri et al., 1996). The states of T are identified as pairs of a state of T_1 and a state of T_2. A state (q_1, q_2) in $T_1 \circ T_2$ is an initial (final) state if and only if q_1 is an initial (resp. final) state of T_1 and q_2 is an initial (resp. final) state of T_2. The transitions of T are the result of matching a transition of T_1 and a transition of T_2 as follows: (q_1, a, b, w_1, q_1') and (q_2, b, c, w_2, q_2') produce the transition

$$((q_1, q_2), a, c, w_1 + w_2, (q_1', q_2')) \tag{4}$$

in T. The efficiency of this algorithm was critical to that of our unit selection system. Thus, we designed an improved composition that we will describe later. Figure 1(c) gives the resulting of the composition of the weighted transducers given figure 2(a) and (b).

3.2 Language Model Weighted Transducer

The n-gram statistical language model we construct for unit sequences can be represented by a weighted automaton G which assigns to each sequence u its log-likelihood:

$$[\![G]\!](u) = -\log(P(u)). \tag{5}$$

according to our probability estimate P. Since a unit sequence u uniquely determines the corresponding halfphone sequence x, the n-gram statistical model equivalently defines a model of the joint distribution of $P(x, u)$. G can be augmented to define a weighted transducer $\hat{G}$ assigning to pairs (x, u) their log-likelihoods. For any halfphone sequence x and unit sequence u, we define $\hat{G}$ by:

$$[\![\hat{G}]\!](x, u) = -\log P(u) \tag{6}$$

The weighted transducer $\hat{G}$ can be used to generate all the unit sequences corresponding to a specific halfphone sequence given by a finite automaton p, using composition: $p \circ \hat{G}$. In our case, we also wish to use the language model transducer $\hat{G}$ to limit the number of candidate unit sequences considered. We will do that by giving a strong precedence to n-grams of units that occurred in the training corpus (see Section 4.2).

Example Figure 2(a) shows the bigram model G estimated from the following corpus:

```
<s> u1 u2 u1 u2 </s>
<s> u1 u3 </s>
<s> u1 u3 u1 u2 </s>
```

where $\langle$`s`$\rangle$ and $\langle$`/s`$\rangle$ are the symbols marking the start and the end of an utterance. When the unit u_1 is associated to the halfphone p_1 and both units u_1 and u_2 are associated to the halfphone p_2, the corresponding weighted halfphone-to-unit transducer $\hat{G}$ is the one shown in Figure 2(b).

3.3 Unit Selection with Weighted Finite-State Transducers

From each sequence $f = f_1 \dots f_n$ of feature vectors specified by the text analysis frontend, we can straightforwardly derive the halfphone sequence to be synthesized and represent it by a finite automaton p, since the first component of each feature vector f_i is the corresponding halfphone. Let W be the weighted automaton obtained by composition of p with $\hat{G}$ and projection on the output:

$$W = \Pi_2(p \circ \hat{G}) \tag{7}$$

W represents the set of candidate unit sequences with their respective grammar costs. We can then use a speech recognition decoder to search for the best sequence u since W can be thought of as the

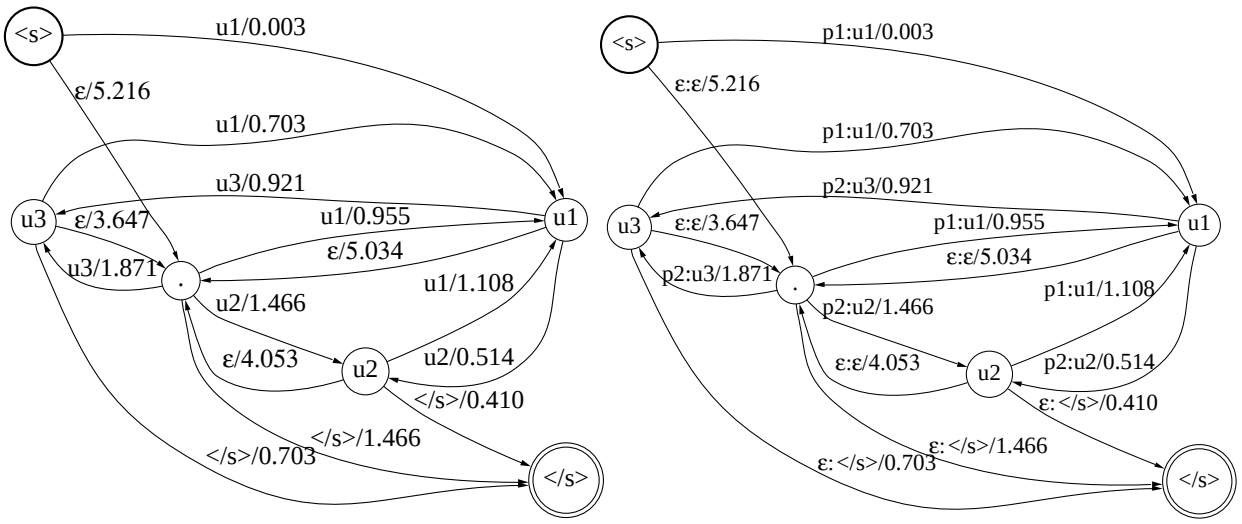

Figure 2: (a) n-gram language model G for unit sequences. (b) Corresponding halfphone-to-unit weighted transducer $\hat{G}$.

counterpart of a speech recognition transducer, f the equivalent of the acoustic features and C_f the analogue of the acoustic cost. Our decoder uses a standard beam search of W to determine the best path by computing on-the-fly the feature cost between each unit and its corresponding feature vector.

Composition constitutes the most costly operation in this framework. Section 4 presents several of the techniques that we used to speed up that algorithm in the context of unit selection.

4 Algorithms

4.1 Composition with String Potentials

In general, composition may create *non-coaccessible states*, i.e., states that do not admit a path to a final state. These states can be removed after composition using a standard connection (or trimming) algorithm that removes unnecessary states. However, our purpose here is to avoid the creation of such states to save computational time. To that end, we introduce the notion of *string potential* at each state.

Let $i[\pi]$ ($o[\pi]$) be the input (resp. output) label of a path π, and denote by $x \wedge y$ the longest common prefix of two strings x and y. Let q be a state in a weighted transducer. The *input* (*output*) *string potential* of q is defined as the longest common prefix of the input (resp. output) labels of all the paths in T from q to a final state:

$$p_i(q) = \bigwedge_{\pi \in \Pi(q,F)} i[\pi]$$
$$p_o(q) = \bigwedge_{\pi \in \Pi(q,F)} o[\pi]$$

The string potentials of the states of T can be computed using the generic shortest-distance algorithm of (Mohri, 2002) over the string semiring. They can be used in composition in the following way. We will say that two strings x and y are *comparable* if x is a prefix of y or y is a prefix of x.

Let (q_1, q_2) be a state in $T = T_1 \circ T_2$. Note that (q_1, q_2) is a coaccessible state only if the output string potential of q_1 in T_1 and the input string potential of q_2 in T_2 are comparable, i.e., $p_o(q_1)$ is a prefix of $p_i(q_2)$ or $p_i(q_2)$ is a prefix of $p_o(q_1)$. Hence, composition can be modified to create only those states for which the string potentials are compatible.

As an example, state $2 = (1, 5)$ of the transducer $T = T_1 \circ T_2$ in Figure 1 needs not be created since $p_o(1) = bcd$ and $p_i(5) = bca$ are not comparable strings.

The notion of string potentials can be extended to further reduce the number of non-coaccessible states created by composition. The *extended input string potential* of q in T, is denoted by $\bar{p}_i(q)$ and is the set of strings defined by:

$$\bar{p}_i(q) = p_i(q) \cdot \zeta_i(q) \quad (8)$$

where $\zeta_i(q) \subseteq \Sigma$ and is such that for every $\sigma \in \zeta_i(q)$, there exist a path π from q to a final state such that $p_i(q)\sigma$ is a prefix of the input label of π. The *extended output string potential* of q, $\bar{p}_o(q)$, is defined similarly. A state (q_1, q_2) in $T_1 \circ T_2$ is coaccessible only if

$$(\bar{p}_o(q_1) \cdot \Sigma^*) \cap (\bar{p}_i(q_2) \cdot \Sigma^*) \neq \emptyset \qquad (9)$$

Using string potentials helped us substantially improve the efficiency of composition in unit selection.

4.2 Language Model Transducer – Backoff

As mentioned before, the transducer $\hat{G}$ represents an n-gram backoff model for the joint probability distribution $P(x, u)$. Thus, backoff transitions are used in a standard fashion when $\hat{G}$ is viewed as an automaton over paired sequences (x, u). Since we use $\hat{G}$ as a transducer mapping halfphone sequences to unit sequences to determine the most likely unit sequence u given a halfphone sequence x [1]we need to clarify the use of the backoff transitions in the composition $p \circ \hat{G}$.

Denote by $O(V)$ the set of output labels of a set of transitions V. Then, the correct use derived from the definition of the backoff transitions in the joint model is as follows. At a given state s of $\hat{G}$ and for a given input halfphone a, the outgoing transitions with input a are the transitions V of s with input label a, and for each $b \notin O(V)$, the transition of the first backoff state of s with input label a and output b.

For the purpose of our unit selection system, we had to resort to an approximation. This is because in general, the backoff use just outlined leads to examining, for a given halfphone, the set of all units possible at each state, which is typically quite large.[2] Instead, we restricted the inspection of the backoff states in the following way within the composition $p \circ \hat{G}$. A state s_1 in p corresponds in the composed transducer $p \circ \hat{G}$ to a set of states (s_1, s_2), $s_2 \in S_2$, where S_2 is a subset of the states of $\hat{G}$. When computing the outgoing transitions of the states in (s_1, s_2) with input label a, the backoff transitions of a state s_2 are inspected if and only if none of the states in S_2 has an outgoing transition with input label a.

[1]This corresponds to the conditional probability $P(u|x) = P(x, u)/P(x)$.

[2]Note that more generally the *vocabulary* size of our statistical language models, about 400,000, is quite large compared to the usual word-based models.

4.3 Language Model Transducer – Shrinking

A classical algorithm for reducing the size of an n-gram language model is shrinking using the entropy-based method of (Stolcke, 1998) or the weighted difference method (Seymore and Rosenfeld, 1996), both quite similar in practice. In our experiments, we used a modified version of the weighted difference method. Let w be a unit and let h be its conditioning history within the n-gram model. For a given shrink factor γ, the transition corresponding to the n-gram hw is removed from the weighted automaton if:

$$\log(\widetilde{P}(w|h)) - \log(\alpha_h \widetilde{P}(w|h')) \leq \frac{\gamma}{c(hw)} \qquad (10)$$

where h' is the backoff sequence associated with h. Thus, a higher-order n-gram hw is pruned when it does not provide a probability estimate significantly different from the corresponding lower-order n-gram sequence $h'w$.

This standard shrinking method needs to be modified to be used in the case of our halfphone-to-unit weighted transducer model with the restriction on the traversal of the backoff transitions described in the previous section. The shrinking methods must take into account all the transitions sharing the same input label at the state identified with h and its backoff state h'. Thus, at each state identified with h in $\hat{G}$, a transition with input label x is pruned when the following condition holds:

$$\sum_{w \in X_h^x} \log(\widetilde{P}(w|h)) \; - \sum_{w \in X_{h'}^x} \log(\alpha_h \widetilde{P}(w|h')) \leq \frac{\gamma}{c(hw)}$$

where h' is the backoff sequence associate with h and X_k^x is the set of output labels of all the outgoing transitions with input label x of the state identified with k.

5 Experimental results

We used the AT&T Natural Voices Product speech synthesis system to synthesize 107,987 AP news articles, generating a large corpus of 8,731,662 unit sequences representing a total of 415,227,388 units. We used this corpus to build several n-gram Katz backoff language models with $n = 2$ or 3. Table 1 gives the size of the resulting language model weighted automata. These language models were built using the GRM Library (Allauzen et al., 2004).

We evaluated these models by using them to synthesize an AP news article of 1,000 words, corresponding to 8250 units or 6 minutes of synthesized speech. Table 2 gives the unit selection time (in seconds) taken by our new system to synthesize this AP

Model	No. of states	No. of transitions
2-gram, unshrunken	293,935	5,003,336
3-gram, unshrunken	4,709,404	19,027,244
3-gram, $\gamma = -4$	2,967,472	14,223,284
3-gram, $\gamma = -1$	2,060,031	12,133,965
3-gram, $\gamma = 0$	1,681,233	10,217,164
3-gram, $\gamma = 1$	1,370,220	9,146,797
3-gram, $\gamma = 4$	934,914	7,844,250

Table 1: Size of the stochastic language models for different n-gram order and shrinking factor.

Model	composition	search	total time
baseline system	-	-	4.5s
2-gram, unshrunken	2.9s	1.0s	3.9s
3-gram, unshrunken	1.2s	0.5s	1.7s
3-gram, $\gamma = -4$	1.3s	0.5s	1.8s
3-gram, $\gamma = -1$	1.5s	0.5s	2.0s
3-gram, $\gamma = 0$	1.7s	0.5s	2.2s
3-gram, $\gamma = 1$	2.1s	0.6s	2.7s
3-gram, $\gamma = 4$	2.7s	0.9s	3.6s

Table 2: Computation time for each unit selection system when used to synthesize the same AP news article.

news article. Experiments were run on a 1GHz Pentium III processor with 256KB of cache and 2GB of memory. The baseline system mentioned in this table is the AT&T Natural Voices Product which was also used to generate our training corpus using the concatenation cost caching method from (Beutnagel et al., 1999b). For the new system, both the computation times due to composition and to the search are displayed. Note that the AT&T Natural Voices Product system was highly optimized for speed. In our new systems, the standard research software libraries already mentioned were used. The search was performed using the standard speech recognition Viterbi decoder from the DCD library (Allauzen et al., 2003). With a trigram language model, our new statistical unit selection system was about 2.6 times faster than the baseline system.

A formal test using the standard mean of opinion score (MOS) was used to compare the quality of the high-quality AT&T Natural Voices Product synthesizer and that of the synthesizers based on our new unit selection system with shrunken and unshrunken trigram language models. In such tests, several listeners are asked to rank the quality of each utterance from 1 (worst score) to 5 (best). The MOS results of the three systems with 60 utterances tested by 21 listeners are reported in Table 3 with their corresponding standard error. The difference of scores between the three systems is not statistically significant (first column), in particular, the absolute difference between the two best systems is less than $.1$.

Model	raw score	normalized score
baseline system	$3.54 \pm .20$	$3.09 \pm .22$
3-gram, unshrunken	$3.45 \pm .20$	$2.98 \pm .21$
3-gram, $\gamma = -1$	$3.40 \pm .20$	$2.93 \pm .22$

Table 3: Quality testing results: we report for each system, the mean and standard error of the raw and the listener-normalized scores.

Different listeners may rank utterances in different ways. Some may choose the full range of scores (1–5) to rank each utterance, others may select a smaller range near 5, near 3, or some other range. To factor out such possible discrepancies in ranking, we also computed the listener-normalized scores (second column of the table). This was done for each listener by removing the average score over the full set of utterances, dividing it by the standard deviation, and by centering it around 3. The results show that the difference between the normalized scores of the three systems is not significantly different. Thus, the MOS results show that the three systems have the same quality.

We also measured the similarity of the two best systems by comparing the number of common units they produce for each utterance. On the AP news article already mentioned, more than 75% of the units were common.

6 Conclusion

We introduced a statistical modeling approach to unit selection in speech synthesis. This approach is likely to lead to more accurate unit selection systems based on principled learning algorithms and techniques that radically depart from the heuristic methods used in the traditional systems. Our preliminary experiments using a training corpus generated by the AT&T Natural Voices Product demonstrates that statistical modeling techniques can be used to build a high-quality unit selection system. It also shows other important benefits of this approach: a substantial increase of efficiency and a greater modularity and flexibility.

Acknowledgments

We thank Mark Beutnagel for helping us clarify some of the details of the unit selection system in the AT&T Natural Voices Product speech synthesizer. Mark also generated the training corpora and set up the listening test used in our experiments.

We also acknowledge discussions with Brian Roark about various statistical language modeling topics in the context of unit selection.

References

Cyril Allauzen, Mehryar Mohri, and Michael Riley. 2003. DCD Library - Decoder Library, software collection for decoding and related functions. In *AT&T Labs - Research*. http://www.research.att.com/sw/tools/dcd.

Cyril Allauzen, Mehryar Mohri, and Brian Roark. 2004. A General Weighted Grammar Library. In *Proceedings of the Ninth International Conference on Automata (CIAA 2004)*, Kingston, Ontario, Canada, July. http://www.research.att.com/sw/tools/grm.

Jean Berstel. 1979. *Transductions and Context-Free Languages*. Teubner Studienbucher: Stuttgart.

Mark Beutnagel, Alistair Conkie, Juergen Schroeter, and Yannis Stylianou. 1999a. The AT&T Next-Gen system. In *Proceedings of the Joint Meeting of ASA, EAA and DAGA*, pages 18–24, Berlin, Germany.

Mark Beutnagel, Mehryar Mohri, and Michael Riley. 1999b. Rapid unit selection from a large speech corpus for concatenative speech synthesis. In *Proceedings of Eurospeech*, volume 2, pages 607–610.

Ivan Bulyko and Mari Ostendorf. 2001. Unit selection for speech synthesis using splicing costs with weighted finite-state trasnducers. In *Proceedings of Eurospeech*, volume 2, pages 987–990.

Alistair Conkie, Mark Beutnagel, Ann Syrdal, and Philip Brown. 2000. Preselection of candidate units in a unit selection-based text-to-speech synthesis system. In *Proceedings of ICSLP*, volume 3, pages 314–317.

Samuel Eilenberg. 1974. *Automata, Languages and Machines*, volume A. Academic Press.

Andrew Hunt and Alan Black. 1996. Unit selection in a concatenative speech synthesis system. In *Proceedings of ICASSP'96*, volume 1, pages 373–376, Atlanta, GA.

Frederick Jelinek. 1976. Continuous speech recognition by statistical methods. *IEEE Proceedings*, 64(4):532–556.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustic, Speech, and Signal Processing*, 35(3):400–401.

Werner Kuich and Arto Salomaa. 1986. *Semirings, Automata, Languages*. Number 5 in EATCS Monographs on Theoretical Computer Science. Springer-Verlag, Berlin, Germany.

Mehryar Mohri, Fernando C. N. Pereira, and Michael Riley. 1996. Weighted automata in text and speech processing. In *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI 1996), Workshop on Extended finite state models of language, Budapest, Hungary*. John Wiley and Sons, Chichester.

Mehryar Mohri, Fernando C. N. Pereira, and Michael Riley. 2000. The Design Principles of a Weighted Finite-State Transducer Library. *Theoretical Computer Science*, 231(1):17–32. http://www.research.att.com/sw/tools/fsm.

Mehryar Mohri. 2002. Semiring Frameworks and Algorithms for Shortest-Distance Problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321–350.

Mehryar Mohri. 2004. Weighted Finite-State Transducer Algorithms: An Overview. In Carlos Martín-Vide, Victor Mitrana, and Gheorghe Paun, editors, *Formal Languages and Applications*, volume 148, VIII, 620 p. Springer, Berlin.

Eric Moulines and Francis Charpentier. 1990. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication*, 9(5-6):453–467.

Fernando C. N. Pereira and Michael D. Riley. 1997. Speech Recognition by Composition of Weighted Finite Automata. In *Finite-State Language Processing*, pages 431–453. MIT Press.

Arto Salomaa and Matti Soittola. 1978. *Automata-Theoretic Aspects of Formal Power Series*. Springer-Verlag: New York.

Kristie Seymore and Ronald Rosenfeld. 1996. Scalable backoff language models. In *Proceedings of ICSLP*, volume 1, pages 232–235, Philadelphia, Pennsylvania.

Andreas Stolcke. 1998. Entropy-based pruning of backoff language models. In *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274.

Yannis Stylianou, Thierry Dutoit, and Juergen Schroeter. 1997. Diphone conactenation using a harmonic plus noise model of speech. In *Proceedings of Eurospeech*.

Jon Yi, James Glass, and Lee Hetherington. 2000. A flexible scalable finite-state transducer architecture for corpus-based concatenative speech synthesis. In *Proceedings of ICSLP*, volume 3, pages 322–325.