Automatic feature selection from a large number of features for phone duration prediction

Gabriel Webster, Sabine Buchholz, Javier Latorre

Toshiba Research Europe, Cambridge Research Laboratory, Cambridge, United Kingdom

{gabriel.webster,sabine.buchholz,javier.latorre}@crl.toshiba.co.uk

Abstract

The present research investigates automatic feature selection for phone duration prediction for computer text-to-speech (TTS), selecting from a large set of 242 candidate features. Two methods for avoiding overfitting the training data are evaluated. Experiments with an American English voice corpus show that automatic feature selection using *n*-fold cross validation combined with a simple per-feature improvement threshold was able to achieve a duration prediction accuracy of 22.5 ms RMSE, a relative error rate reduction of 7.8% over a manually selected baseline feature set.

Index Terms: speech synthesis, phone duration prediction, automatic feature selection, feature set

1. Introduction

The choice of feature set for machine learning (ML) algorithms is a critical component of their performance. For phone duration prediction for text-to-speech (TTS) systems, much research has been conducted on the choice of algorithm [1-6], but less research has investigated the optimal input feature sets to these machine learners [3,7,8,9]. One powerful approach for choosing an optimal feature set is *automatic feature selection*, in which a subset of a set of candidate features is chosen automatically. For phone duration prediction, a challenge for any selection method is the large number of potential features that could be tried, due to the large number of potentially relevant linguistic domains (phone, syllable, word, etc.). This makes manual feature selection time-consuming and increases the possibility of overfitting on the training data. Therefore, in this paper, automatic feature selection from a very large number of candidate features is investigated.

Most papers that have investigated how to choose an optimal feature set for duration prediction focus either on manually selecting an optimal feature set [3,9], or on analyzing the effectiveness of individual features in a manually chosen feature set [7], without actually optimizing the set. Since these methods are not automatic, they are time-consuming, and indeed with a very large set of candidate features the vast number of possible combinations makes it extremely difficult to manually optimize the feature set in any meaningful way. Furthermore, a feature set chosen for one corpus will not necessarily perform as well on a corpus of a different speaker, language or size.

Automatic feature selection overcomes these problems. It can be applied to any voice in any language, of any corpus size, and it chooses the feature set that best captures the durational characteristics of that voice. Furthermore, because it is completely automatic, it can try a large number of possible features sets in a reasonable amount of time.

True automatic feature selection for phone duration prediction was previously investigated by Ozturk and Ciloglu [8]. Unfortunately, the feature selection in the paper selects from a set of only 17 possible features, and it suffers from the serious methodological error of selecting the feature set to directly optimize performance on the test data. There was no unseen data on which to test the final selected feature set, and thus no way to know if the selected feature set was overfitted to the test set.

In fact, avoiding overfitting on the training data is arguably the primary challenge of automatic feature selection. Furthermore, selecting from a large number of features increases the chance that a feature will improve performance on a development set (a secondary test set used to evaluate intermediate feature sets) completely by chance, rather than because it truly correlates with the dependent variable.

The goal of this paper, then, is to establish how well automatic feature selection from a large number of candidate features can perform for phone duration prediction. Accomplishing this requires the secondary goal of investigating techniques to avoid overfitting on the training data during selection.

2. Method

2.1. Data

The data used for these experiments was a speech corpus of a female speaker of American English recorded by Toshiba Research Europe. The corpus was 5.5 hours long. The corpus was transcribed with manually corrected phonetic transcriptions in a proprietary 43-symbol phone set which was conventional except in that plosive closures were transcribed separately from plosive releases and clustered together for the purposes of duration modeling. The total number of phones in the corpus was 125,543.

The data was divided into 80% training, 10% development, and 10% testing. All model training, including automatic feature selection, was performed using the training and development data only. The test data was only used to determine the final accuracy of the final optimized feature sets identified during feature selection.

2.2. Algorithm

Multiple linear regression (MLR) was used as the machine learning algorithm for all experiments. MLR assumes that a dependent variable can be modeled as a linear combination of independent variables (features) multiplied by appropriately estimated coefficients. The general model form is

$$y=\beta_1 x_1+\ldots+\beta_p x_p+e \qquad (1)$$

where y is the dependent variable, the x are the independent variables, the β are the estimated coefficients, p is the number of independent variables, and e is an error term representing the noise in the model. An MLR model is trained by estimating values for each coefficient, either by matrix methods or by gradient descent.

Symbolic (non-numeric) features are a special case in MLR. Since symbolic feature values cannot be directly multiplied by a coefficient, MLR estimates a coefficient for each possible feature value of these features. Then during prediction, only the coefficient that corresponds to the input feature value is added to the final prediction. Internally, this is done by converting a feature with *n* possible values into *n* Boolean features, and then assigning *True* to the feature corresponding to the input feature value and *False* to the other features.

For the current experiments, a variant of MLR in which true continuous features are not modeled was used [10]. Instead, individual coefficients are estimated for each value of a semantically continuous variable. This enables nonlinear modeling of semantically continuous variables, at the cost of increasing the number of parameters that must be estimated and thus the risk of data sparsity. It was due to this risk that the automatic feature value clustering method described in section 2.4 was used.

Because the duration properties of phones vary considerably from one phone to the next, separate regression models were trained for each phone, and automatic feature selection was performed separately for each regression model.

Because MLR does not model interactions between features, such interactions cannot cause stepwise automatic model selection methods to get stuck in local minima. Thus, for the automatic feature selection, a simple greedy forward selection method was chosen. In this method, each feature was first evaluated individually. Then, the features were sorted in descending order of accuracy and each was added in turn to the feature set. If the feature improved accuracy, then it was kept in the feature set; otherwise, it was discarded.

To prevent overfitting on the training data, two different techniques were tried. The first was the Bayesian Information Criterion (BIC) [11], where a candidate feature had to improve the BIC of the training data in order to be selected into the final feature set. The second method was *n*-fold cross validation, *n=8*, combined with a simple, constant minimum improvement threshold. This threshold required each candidate feature to improve the mean accuracy on the cross-validation test sets by at least a certain fixed amount in order to be selected. The value of the threshold was determined empirically by choosing several possible threshold values, performing automatic feature selection with cross validation on the training set using each value, and then evaluating the accuracy of the resulting feature sets on the development set. The threshold which resulted in the highest development set accuracy was used as the final improvement threshold.

2.3. Feature set

A total of 242 features were used as the set available for feature selection to choose from. (We do not claim that the set contains every feature that could potentially be tried; it is simply a very large set.) Due to the large number of features, a very compact representation is necessary for describing them here. If a number in parentheses (*n*) follows the feature name, then the feature was actually made available as a window of *n* features centered on the unit corresponding to the current phone (values of *n* were chosen arbitrarily). An asterisk following the window size means that the central feature of that window always had the same value for each model (e.g. phone label) and was not used. Note that the information encoded by some features overlaps; this allows feature selection to choose the encoding(s) which it finds most useful.

Feature values were calculated using part of speech (POS), grammatical role and distance, prosodic break and pitch accent values that were predicted (by conventional ML methods), rather than using manually corrected gold standard values (the ML models were trained on gold standard values). This was done firstly to eliminate training/synthesis mismatch (since during synthesis, predicted values are input to phone duration prediction), and secondly because it results in a realistic judgement of such features' usefulness.[1]

The complete feature set was as follows, with definitions when necessary:

- *punctuation* (5): The punctuation mark after the current word, if any
- *part of speech (POS)* (5): The predicted POS of the current word
- *contentFunction* (5): Whether the current word is a content or function word, based on *POS*
- *extendedContentFunction* (5): An "extended" content-function tag, as defined by Busser et al. [12], which takes the value of the POS if the word is a content word, or the actual word otherwise, but with uncommon function words clustered to "other"
- *role* (5): The grammatical role, such as "subject" or "object", predicted for the current word
- *numWordsToHead* (5): The predicted distance between the current word and its grammatical head
- *lowestSpanDependencyRole* (5): The grammatical role of the lowest, i.e. shortest, dependency link that spans the juncture between the current and following words, as defined by Hunt [13]. For example, if the lowest spanning role is "subject", then the words on the left side of the juncture belong to the subject while the words on the right belong to the predicate
- *numWordsInLowestSpanSubstructure* (5): The number of words in the dependency substructure which is headed by the shortest dependency link that spans the juncture following the current word. Depending on the direction of the link, this corresponds to the size of the constituent to the left or the right of the juncture
- *prosodicBreakFlag* (5): Whether or not a prosodic break is predicted to follow the current word
- *pauseFlag* (5): Whether or not a pause follows the current word
- *pitchAccent* (5): Predicted pitch accent value (accented or deaccented)
- *phoneLabel* (9*): The name of the current phone
- *phoneType* (9*): One of "LongVowel", "ShortVowel", "Diphthong", "VoicedPlosive", "UnvoicedPlosive",

[1] It might be argued that training phone duration prediction on predicted values confounds the accuracy of duration prediction with the accuracy of predicting values in previous modules. However, it was felt that the accuracy of predicting these values was fairly close to the state of the art (e.g., 0.75 F-score for Boolean prosodic break prediction; 90% accuracy for two-level pitch accent prediction), and that overall it was more important to select feature sets that reflected the true usefulness of candidate features in actual synthesis. Furthermore, using non-predicted values opens up the possibility of introducing candidate features that are useful but difficult to predict accurately, such as "hasContrastiveFocus". Automatic feature selection would trivially select these features, but this wouldn't reveal anything about improving the quality of actual TTS systems.

"OtherVoicedConsonant", "OtherUnvoicedConsonant", "Closure"

- *phone{IsVoiced|Place|Manner|VoicingAndManner|Height|Backness|Tenseness|Roundness}* (9*): Linguistic properties of current phone
- *sylVowelLabel* (5): Label of vowel of current syllable
- *phoneIsStressed* (9): Whether the current phone is a lexically stressed vowel
- *sylRelativeToStress*: Whether the current syllable is before, after, or the same as the lexically stressed syllable of the current word
- *sylIsStressed* (5): Whether the current syllable has lexical stress
- *sylHasPitchAccent* (5): Whether the predicted pitch accent of the current syllable is "accented" or "deaccented"
- *numPhonesToSylNucleus*: Number of phones between current phone and vowel of current syllable; negative if the current phone is after the vowel of the current syllable
- *numPhonesToSyl{Begin|End}*: Number of phones between the current phone and the beginning/end of the current syllable
- *num{Phones|Syls}ToWord{Begin|End}*
- *num{Phones|Syls|Words|Breaks|Pauses|Puncs}ToSent{Begin|End}*: "Break" is prosodic break; "Punc" is punctuation mark
- *num{Phones|Syls}To{Previous|Next}{Stress|Break|PitchAccent|Pause|Punc}*
- *numWordsTo{Previous|Next}{ContentWord|Break|PitchAccent|Pause|Punc}*
- *numContentWordsTo{Previous|Next}Pause*
- *numBreaksTo{Previous|Next}{Pause|Punc}*
- *numPausesTo{Previous|Next}Punc*
- *numPhonesInThisSyl{Onset|Coda}*: The number of phones in the onset/coda of the current syllable (regardless of where in the current syllable the current phone is located)
- *numPhonesInThisSyl* (5)
- *num{Phones|Syls}InThisWord* (3)
- *num{Phones|Syls|Words}InThis{Chunk|PausePhrase|Sent}*: "Chunk" is one or more consecutive words delimited by prosodic breaks (or sentence boundaries); "Pause phrase" is the same but delimited by pauses
- *numChunksInThis{PausePhrase|Sent}*
- *numPausesInThisSent*

2.4. Automatic feature value clustering

One drawback of MLR is that many instances of each feature value must exist in the training data in order to reliably estimate coefficients for the feature. If the number of instances is too small, then coefficients that overfit the training data may be estimated. For example, if a feature value appears exactly once in the training data, then a coefficient can be estimated that simply reduces the error of that particular training instance to zero. This overfitting problem might be exacerbated by model selection: features containing infrequent features values might tend to be chosen, because these might reduce the training error more significantly (albeit only by overfitting the training data).

To help avoid this problem, feature values were automatically clustered prior to training (manual clustering was impractical due the large number of features). For discrete features, it is difficult to determine *a priori* which feature values are more closely related than others. For this reason, a simple frequency-based clustering method was chosen: for each feature, the least frequent value was iteratively clustered with the next least frequent value until all clustered values had at least 10 instances. For semantically continuous features (recalling that the version of MLR used does not support true continuous features), the least frequent value was iteratively clustered with the value that is numerically the closest on the side closer to zero (that is, smaller than a positive value, and larger than a negative value), until all clustered values had at least 10 instances.

3. Results

Firstly, a baseline feature set was defined. This was chosen to be a manually determined feature set that was extensively tuned on the same speech corpus as that used for the experiments, and consisted of the following 19 features:

- *numPhonesToSylNucleus*
- *numPhonesToSyl{Begin|End}*
- *numSylsToNext{Stress|Break}*
- *numSylsTo{Previous|Next}Pause*
- *numSylsToWord{Begin|End}*
- *POS*
- *pitchAccent*
- *{previous1|next1|next2}PhoneType*
- *sylRelativeToStress*
- *numPhonesInThisSyl{Onset|Coda}*
- *numPhonesInThisSylMaxN*
- *numSylsInThisWordMaxN*

Duration prediction accuracy on the test set using this baseline feature set was 24.4 ms RMSE.

Next, the accuracy of automatic feature selection using BIC was evaluated. Because the goodness-of-fit measurement of BIC is calculated on the training data itself, the training and development data were pooled together for automatic feature selection. The resulting test set accuracy was 23.3 ms RMSE.

Finally, automatic feature selection with *n*-fold cross validation with and without the minimum improvement threshold was carried out. The best threshold was found to be 0.06 ms RMSE. Then, the feature set selected using the best improvement threshold was evaluated on the test data, resulting in an accuracy of 22.5 ms RMSE. Without the threshold, accuracy was 22.7 ms RMSE. All results are summarized in Table 1, along with the mean number of features selected for each phone model.

Model	Features per phone	RMSE accuracy	RERR over baseline
Baseline	19	24.4 ms	n/a
BIC	10	23.3 ms	4.5%
Xval, no thresh.	40	22.7 ms	7.0%
Xval, thresh.	16	22.5 ms	7.8%

Table 1: Phone duration prediction accuracies

The results show that all methods of preventing overfitting offer an improvement over the baseline manually selected feature set, with *n*-fold cross validation with the improvement threshold giving the overall best accuracy, representing a 7.8% relative error rate reduction (RERR) over the baseline model. The improvement over cross validation with no threshold, along with the large reduction in number of features selected (60%), suggests that the threshold was helpful in preventing overfitting on the training data.

4. Discussion

An analysis of the most accurate automatic feature selection method, *n*-fold cross validation with an improvement threshold, was performed to better understand the nature of the features that were selected. Firstly, since automatic feature selection was performed separately for each phone model, a metric was designed to calculate the most frequently selected features across all models. For this metric, a score of 1.0 was assigned to the first-selected feature for each model, and then linearly decreasing scores were assigned to subsequently chosen features such that the final feature chosen for each phone model was assigned a score of $1/l_p$, where l_p is the total number of features selected for phone *p*. For example, if four features were selected for one phone model, the features would receive scores of 1.0, 0.75, 0.5, and 0.25, in the order that the features were selected. All features not selected for a model were assigned a score of 0.0. Then, for each feature, a final weighted average score across all models was calculated, with the weights equal to the number of test cases for each phone, in order to give more importance to features selected for more frequent phones. So in these final scores, a feature that was selected first for every phone model would receive a score of 1.0, while a feature that was never chosen would receive a score of 0.0. The 20 features with the highest scores given this metric are listed in Table 2.

	Feature	Score
1	next1PhoneLabel	0.76
2	previous1PhoneLabel	0.59
3	next2PhoneLabel	0.55
4	numPhonesToNextPause	0.47
5	numPhonesToNextBreak	0.45
6	numPhonesToWordBegin	0.33
7	numPhonesToNextPitchAccent	0.30
8	lowestSpanDependencyRole	0.23
9	numSylsToNextBreak	0.23
10	numPhonesToWordEnd	0.17
11	extendedContentFunction	0.15
12	numSylsToNextPause	0.14
13	POS	0.14
14	numPhonesToSylBegin	0.14
15	sylHasPitchAccent	0.13
16	next1PhoneIsStressed	0.13
17	numPhonesToNextPunc	0.12
18	previous2PhoneLabel	0.10
19	sylVowelLabel	0.10
20	sylRelativeToStress	0.10

Table 2: Most frequently selected features

The most important features in this list appear to be mainly of two different types. Firstly, and most importantly, are the phone context features *{previous|next}{1|2}PhoneLabel*, which include the overall top three features. Secondly in importance seem to be features about the distance to following boundaries, encoded by the six features *num{Phones|syls}-ToNext{Break|Pause}* and *numPhonesTo{WordEnd|Next-Punc}*. Pairs of features differing only in unit size (phone or syllable) were selected twice, suggesting that counting in different sized units provides complementary information.

Seven of the top 20 features contain predicted feature values: *numPhonesToNext{Break|PitchAccent}*, *numSylsToNextBreak*, *sylHasPitchAccent*, *lowestSpanDependencyRole*, *POS*, and *extendedContextFunction*. This demonstrates that predicted features from other modules need not be perfectly accurate in order to be important for improving duration prediction accuracy, as long as they are accurate enough.

Feature selection frequency scores were also calculated separately for vowels and consonants. The most salient difference was that the features *POS* and *extended-ContentFunction* were ranked 10 and 11 for vowels, but were not within the top 20 features for consonants. This suggests that these features were primarily being used to model the fact that function words are often pronounced with shorter phone durations than content words, and that this difference is reflected primarily in vowel phones.

5. Conclusion

The experiments reported in this paper demonstrate that automatic feature selection from a very large number of features offers a clear objective improvement in phone duration prediction over a manually selected baseline feature set. *N*-fold cross validation combined with a simple improvement threshold is an effective way of preventing the automatic feature selection from overfitting the training data. Future research may include developing a more sophisticated improvement threshold, such as phone-specific thresholds or a threshold dependent on the number of model parameters.

6. References

[1] M. Riley, "Tree-based modelling for speech synthesis", in *Talking Machines: Theories, Models and Designs*, G. Bailly, C. Benoit, and T.R. Sawallis, Eds. Elsevier, Amsterdam, Netherlands, 1992, pp. 265–273.

[2] K. Takeda, Y. Sagisaka, and H. Kuwabara, "On sentence-level factors governing segmental duration in Japanese", *Journal of Acoustic Society of America*, vol. 86, no. 6, pp. 2081–2087, 1989.

[3] J.P.H. van Santen, "Assignment of segmental duration in text-to-speech synthesis", *Computer Speech and Language*, vol. 8, pp. 95-128, 1994.

[4] R. Cordoba, J.M. Montero, J. Gutierrez-Ariola, and J.M. Pardo, "Duration modeling in a restricted-domain female voice synthesis in Spanish using neural networks", in *Proc. ICASSP*, Salt Lake City, 2001.

[5] W.N. Campbell, "Syllable based segment duration", in *Talking Machines: Theories, Models and Designs*, G. Bailly, C. Benoit, and T.R. Sawallis, Eds. Elsevier, Amsterdam, Netherlands, 1992, pp. 211–224.

[6] J.T. Chien and C.H. Huang, "Bayesian Learning of Speech Duration Models", *IEEE Trans. Speech and Audio Proc.*, vol. 11, no. 6, pp. 558-567, 2003.

[7] S. Breuer, K. Francuzik, and G. Demenko, "Analysis of Polish Segmental Duration with CART", in *Proc. Speech Prosody*, Dresden, 2006.

[8] O. Ozturk and T. Ciloglu, "Segmental Duration Modelling in Turkish", in *Text, Speech and Dialogue*, P. Sojka, I. Kopecek, and K. Pala, Eds. Springer-Verlag, 2006, pp. 669-676.

[9] M. Mihkla, "Modelling speech temporal structure for Estonian text-to-speech synthesis: Feature selection", *TRAMES*, vol. 11, no. 2, pp. 284-298, 2007.

[10] C. Hayashi, "On the quantification of qualitative data from the mathematico-statistical point of view (an approach for applying this method to the parole prediction)," *Annals of the Institute of Statistical Mathematics*, pp. 35-47, Dec. 1950.

[11] G.E. Schwarz, "Estimating the dimension of a model", *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[12] B. Busser, W. Daelemans, and A. van den Bosch, "Predicting phrase breaks with memory-based learning", in *Fourth ISCA ITRW on Speech Synthesis*, Perthshire, Scotland, 2001.

[13] A.J. Hunt, "Improving Speech Understanding Through Integration of Prosody And Syntax", in *Proc. 7th Australian Joint Conf. on Artificial Intelligence*, Armidale, 1994.