Minimum Error Rate Training for Phrasing in Speech Synthesis

Alok Parlikar and Alan W Black

Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA. USA
{aup, awb} @cs.cmu.edu

Abstract

Phrase break prediction models in speech synthesis are classifiers that predict whether or not each word boundary is a prosodic break. These classifiers are generally trained to optimize the likelihood of prediction, and their performance is evaluated in terms of classification accuracy. We propose a minimum error rate training method for phrase break prediction. We combine multiple phrasing models into a log-linear framework and optimize the system directly to the quality of break prediction, as measured by the F-measure. We show that this method significantly improves our phrasing models. We also show how this framework allows us to design a knob that can be tweaked to increase or decrease the number of phrase breaks at synthesis time.

Index Terms: Speech Synthesis, Phrasing

1. Introduction

Phrase break prediction (phrasing) is an important prosodic step during speech synthesis. Other prosody models depend on phrasing decisions, and hence appropriate phrase breaks are critical to overall naturalness of synthetic speech. The problem of phrasing can be thought of as a classification problem: given some text, we want to classify each word boundary as being a phrase break or not. In terms of the TOBI[1] systems for prosody annotation, phrasing classifiers are typically trained to predict levels 1, 3, and 4. Phrasing classifiers are often trained using standard corpora: e.g., the Festival[2] system uses a model[3] trained on the MARSEC[4] data for English voices. If manually annotated data is not available, phrasing models can be trained by force-aligning the speech and text data available for building synthetic voices[5].

In practice, phrasing models can be decomposed into two disparate models: (i) A model that takes local context of a word boundary into account to decide how likely a break is, and (ii) A model that takes longer context of previously generated breaks to decide how frequently breaks should be generated. These two models can be combined together, such as using the Viterbi method, to decide the optimal sequence of phrase breaks.

This paper discusses two important aspects of phrasing, and attempts to build upon the state of the art: (i) Optimization target for phrasing, and (ii) Phrasing and changes in speaking rate.

Phrasing classifiers are typically trained to maximize the likelihood of break prediction. However, they are then evaluated on the basis of an accuracy measure, such as F-1 score[6]. An objective improvement in F-1 score is also perceived by people in subjective listening tests. In order to make a higher perceptual impact, we aim to optimize our phrasing model directly to the F-1 score.

Phrasing models are usually insensitive to the speaking rate. In natural speech, we observe that fast speech has fewer phrase breaks, and slower speech tends to have more breaks. A phrasing model needs to thus provide a mechanism that allows us to insert more or fewer breaks at synthesis time depending on the speaking rate.

We propose a minimum error rate training approach that provides a solution to both these needs in phrasing. The idea here is to combine multiple phrasing models in a log-linear fashion, and learn weights for different models in order to maximize the F-measure of break prediction. We describe how such a framework not only improves the break prediction, but also offers a “knob” to increase or decrease the number of predicted breaks. We also discuss how this framework can be useful in solving other difficult problems, such as phrasing in the context of the synthesis of disfluent machine translation output.

2. Classic Phrasing (Baseline)

Given a text corpus annotated with breaks, there are several ways of training a phrasing model. Rule based methods[7], or data driven methods using machine learning techniques such as decision trees[8, 9], transformational rule learning[10], z-score models[11], hidden Markov models[3, 12, 13], memory based learning[14], Baysian networks[15], maximum entropy models[16], and neural networks[17] have been successful at the task. In this work, we use the models[3, 5] that have been setup for the Festival speech synthesis system[2].

Festival’s phrasing uses two models: a context model, and a sequence model. If b_i is the probability of a break at the juncture i, C_i is the context of features at juncture i, and B_i represents the context of previous break sequences at juncture i, then we want to estimate $P(b_i|C_i, B_i)$. With the help of the Bayes theorem, and a few independence assumptions, we can derive that

$$P(b_i|C_i, B_i) \propto \frac{P(b_i|B_i) \cdot P(b_i|C_i)}{P(b_i)}.$$

The term $P(b_i|B_i)$ is essentially the language model probability of a given break sequence. We estimate this in Festival using a 7-gram model[3]. The term $P(b_i)$ is the unigram probability of a word boundary being a break. Our context model, $P(b_i|C_i)$, is a grammar based model[5]. This is estimated using a decision tree classifier, that uses word level features, positional features, and syntactic features. Given these models, at synthesis time, our goal is to find the optimal break sequence b^*:

$$b^* = \arg\max_b \prod_i P(b_i|C_i, B_i)$$

$$\therefore b^* = \arg\max_b \prod_i P(b_i|C_i) \cdot P(b_i|B_i)$$

.

In order to find the most likely break sequence, Festival runs a Viterbi search over the possible phrase break sequences. This is schematically shown in Figure 1.

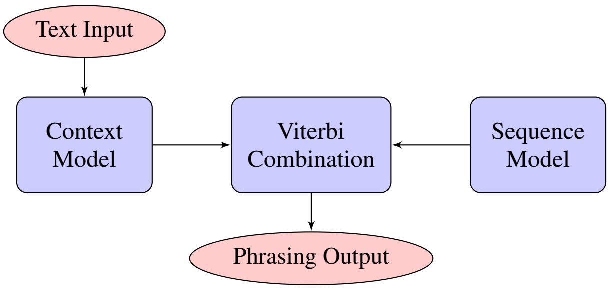

Figure 1: Viterbi Phrasing Strategy in Festival

3. Minimum Error Rate Training

Ideally we want to train our phrasing model such that the end-to-end performance in perceived synthesis quality is optimal. A model trained to maximize the likelihood of phrase breaks makes the simplifying assumption that the final evaluation is based on simply counting the number of wrong decisions made. However, a metric such as F-measure is a little more complex, since it takes both precision and recall into account, and is better correlated to perception than just the accuracy of phrasing. Our goal is to optimize the selection of phrase breaks in order to minimize the error our model makes, as measured by the F-1 score. The idea of using minimum error-rate training (MERT) in phrasing is inspired from its use in the Statistical Machine Translation[18].

Let us assume that we are given a text sequence $\mathbf{t}$, and we want to produce a break sequence $\mathbf{b}$. Among all possible break sequences, we will choose the sequence with the highest probability:

$$\mathbf{b}^* = \arg\max_b P(\mathbf{b}|\mathbf{t}).$$

We directly model the posterior probability $P(\mathbf{b}|\mathbf{t})$ using a log-linear model. In this framework, we have a set of M feature functions, $h_m(\mathbf{b}, \mathbf{t})$. For each feature function, we have a weight w_m. The direct phrasing probability is then given by:

$$P(\mathbf{b}|\mathbf{t}) = \frac{\exp\left(\sum_{m=1}^{M} w_m h_m(\mathbf{b}, \mathbf{t})\right)}{\sum_{b'} \exp\left(\sum_{m=1}^{M} w_m h_m(\mathbf{b}', \mathbf{t})\right)}.$$

The modeling problem here is to define suitable feature functions that capture the relevant properties of the phrasing task. The training problem is to find out suitable weights w_1^M. However, as mentioned before, we want to train the model to minimize error.

We define a held out development corpus D_1^N, of size N, with text sequences T_1^N that has reference break annotations R_1^N. Our goal is to obtain minimum error on this corpus, and a set of k different candidate break sequences, $S_n = \mathbf{b}_{n,1}, \ldots, \mathbf{b}_{n,k}$. That is, for each of the N sentences in the test set, we have k hypotheses of break sequences, and we want to pick the ones that minimize overall error on the test set. Given a set of weights w_1^M, the top-best break sequence $\mathbf{b}_n$ for sentence n is given by:

$$\mathbf{b}_n = \arg\max_{\mathbf{b} \in S_n} \left[\sum_{m=1}^{M} w_m h_m(\mathbf{b}|T_n) \right].$$

For each sentence in the development corpus D_1^N, we can pick the best break sequence given some weights, and then compute the F-measure over these break sequences. The error function E can then be set to negative value of the F-measure. If $E(D_1^N; w_1^M)$ represents the error on the test set given a set of weights, we have:

$$w_1^M* = \arg\min_{w_1^M} \left[E(D_1^N; w_1^M) \right].$$

The optimization criterion here is tricky. Because of the presence of an $\arg\max$ operation within the Error function, we can not compute the gradient of the error, and hence an optimization method such as gradient descent can not be used here. The error surface is not smooth, and has many local minima.

We use the Basin-Hopping algorithm[19] to optimize the error function at hand. This global minimization method has been shown to be extremely efficient for a wide variety of problems, and is especially useful when the error function has many minima separated by large barriers. In particular, we use the implementation of this algorithm within the Python SciPy toolkit.

We use a development corpus, use a randomly initialized weight sequence and produce an n-best list of break sequences. We then run the minimum error rate training over these n-best sequences and learn new weights. We then use the new weights and re-generate an n-best list of break sequences over the development corpus, and run minimum error rate training again. We repeat these iterative process until the final error does not improve across an iteration. After each iteration, we also normalize the weight vector to be of a unit norm.

We use four feature functions $h_m(\mathbf{b}|\mathbf{t})$ in our method. Two of these are the same as the models in the baseline phrasing method: (i) The context model $P(b_i|C_i)$ that looks at the lexical and syntactic context, and (ii) $P(b_i|B_i)$ that looks at the language model probability of the break sequence. In addition, we use another context model, $P(b_i|C_i)$, defined in [3] that looks at the part-of-speech tag context at a word boundary and uses a quadgram Language Model to predict the probability of the word boundary being a break. Finally, we use a break-count feature, that counts the total number of breaks in the predicted break sequence.

4. Experimental Results

We evaluated our method on two synthetic voices trained using the CLUSTERGEN[20] statistical parametric synthesis method: (i) Voice built from about an hour of speech in the F2B corpus within the Boston University Radio News Corpus[21], and (ii) Voice built from two hours of recordings of Jane Austen's books, for Blizzard Challenge 2013 task EH2. We split the corpora into splits of 80-10-10 for training, development and testing.

Our baseline phrasing models were built to be style-specific phrasing models[5] in each case. We trained the proposed model with minimum error rate method on a held out corpus, and used the unseen test partition in the same domain to compare the baseline method to the proposed approach. Table 1 shows the comparison of the models in terms of the F-1 metric[6]. We see that the proposed method yields an improvement over the baseline on both datasets.

5. Phrasing Rate: "Knob"

One requirement of a phrasing model is that it should be flexible to adapt to the speaking rate of a synthesizer. A slow synthesizer should probably mark more word boundaries as breaks, and a faster synthesizer can do away with a few breaks. If the user of a speech synthesis engine demands that 30%, or 60% of the word

Table 1: Objective Evaluation (F-1 measure) of the Proposed MERT Method. Improvements are significant at $p < 0.05$

Voice	Baseline	Proposed
F2B	54.35	58.06
Audiobook	52.87	57.58

boundaries should be breaks, then our phrasing model should be able to meet this requirement. However, this is a tricky constraint. If our training data had splits corresponding to slower and faster speaking styles, we could train individual classifiers and use the appropriate one at synthesis time. But such data is seldom available, and collecting data to train such specific models is difficult. We describe how we use the log-linear framework and MERT mechanism to provide a knob, a continuous number, to vary the number of phrase breaks produced.

One of the features that we used in the log-linear model was simply the number of phrase breaks in a given break sequence. This feature allows us to define a knob to change the number of phrase breaks our model produces.

Intuitively, the break-count feature tries to make sure that the number of breaks produced by our model is reasonably close to the number of breaks in the reference sequences in our development data. Even if we optimize towards the F-measure of break prediction, which itself balances precision and recall of phrasing, having this additional feature means that the weight learned for this feature will produce an optimal number of phrase breaks. If we keep the weights for other features to be the same, and change the weight of the break-count feature, then the search process at synthesis time picks utterances with more or fewer breaks than the optimal. For example, if we subtract a number from the weight of the break count feature, and maximize the log-linear combination, we would produce more breaks. We can vary the value of this weight and measure the number of word boundaries in a development corpus that were breaks. The weight of the break-count feature is thus the knob we can use to tweak the amount of phrasing. Figure 2 shows this curve for the two voices we have. The x-axis shows the value of the knob (i.e., the weight of the break-count feature) and the y-axis shows what percentage of word boundaries in a corpus were predicted as being breaks.

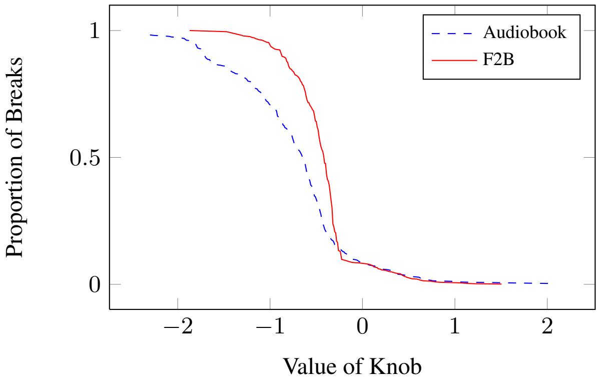

Figure 2: Proportion of phrase breaks generated by varying the log-linear weight of the break-count feature (the knob)

In order to customize the phrasing rate at demand, we need to parameterize the "knob", so that given a particular value of expected proportion of breaks, we can set an appropriate weight for the break-count feature during synthesis. This problem boils down to deriving an equation for the inverse of the function represented in Figure 2. Given a particular phrasing proportion x, we want to find out the value k that our knob should be set to.

To learn the parametric equation of the knob, we use a development corpus and varying values of the weight of the break-count feature to generate data points depicted in Figure 2. We then fit the data automatically to a variety of sigmoidal, trignometric and simple functions and choose the function that best fits the data we have, as measured by the root-mean-squared error of the fit. We used open-source fitting code, pyeq2[22] in this work.

Our empirical analysis shows that for various corpora and phrasing model combinations, the phrasing knob curve can be approximated, well within a RMS tolerance of about 0.15, using a Tangent equation with offset:

$$k = A \cdot \tan\left(\pi \frac{x - C}{W}\right) + O$$

where A (Amplitude), C (center), W (width) and O (offset) are the parameters we learn automatically. For the F2B voice, we obtained

$$k_{f2b} = -0.165 \cdot \tan\left(\pi \frac{x - 0.5073}{1.0772}\right) - 0.4670$$

and for the Audiobook voice, we obtained

$$k_{audiobook} = -0.2682 \cdot \tan\left(\pi \frac{x - 0.5048}{1.072}\right) - 0.8084$$

By tweaking the knob to change the phrasing rate, we deviate from the reference break sequences that we originally used to train our MERT model. This means, by changing the knob, we obtain fewer or more breaks, but at the cost of the F-measure. Of course, since the goal was to insert more or fewer breaks, the penalty in F-measure is not very relevant anymore, but we looked at what the drop in the F-measure looks like. Figure 3 shows how the F-measure changes when we set the expected break proportion to different values. We observe that the F-measure is highest when the knob is set to its original value, as learned from the MERT training.

6. Conclusions and Future Work

We described our method of defining the phrasing problem under a log-linear framework and training the framework with a minimum error rate target, rather than maximum likelihood. We showed that combining features/models related to phrasing using this MERT strategy produces a significant improvement in phrasing accuracy, as measured by the F-1 metric.

We described a break-count feature integral to our MERT model that allows us to define a parametric "knob" to vary the quantity of generated phrase breaks. Once we learn our MERT weights, we can keep all weights to their learned value and vary the weight of the break-count feature to provide this knob. Our empirical evidence shows that the knob can be reasonably approximated with a Tangent function with offset. The combination of using a MERT model and this break-count feature allows a user to specify how many breaks they want, and our model produces the breaks appropriately.

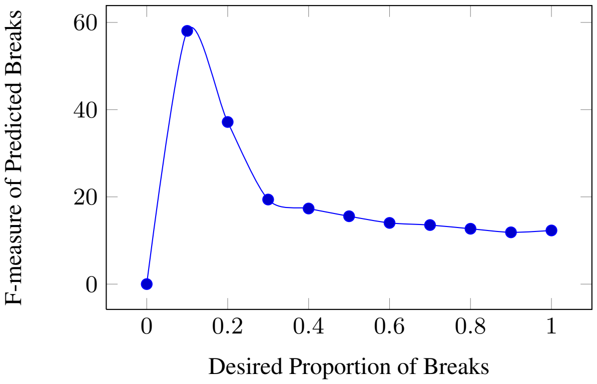

Figure 3: F-measure versus Desired Proportion of Phrase Breaks on the F2B corpus

We intend to explore the benefits of the proposed framework in more details. Particularly, we only used four feature functions in this work and would like to investigate how additional features could help improve the phrasing accuracy even further.

Our model now can vary the phrasing rate at demand. We map a particular phrasing proportion into a knob value. However, users of speech synthesis do not define phrasing rate in terms of the proportion of word boundaries that are breaks. The control we would like them to have would be more quantized: low, medium, high, etc. However, how does a particular category, such as "low" map into the desired proportion of break boundaries? We think this mapping depends on the original proportion of breaks in our reference break sequences. However, we intend to conduct listening tests to discover what grades of phrase breaks people can perceive, and how we can map categories of break levels into numeric proportion values.

The MERT framework that we proposed for phrasing took inspiration from work in Machine Translation. However, this connection actually runs deeper. Text to speech is often used as the final step in speech to speech translation, and we are required to synthesize automatically translated output. [23] has shown that the synthesis of machine translation output is often difficult to understand, and [24] suggests that appropriate phrasing can make it more understandable. We aim to use the MERT framework to incorporate internal machine translation scores of an utterance, that relate to confidence measures of the MT system, into the phrasing model and improve the intelligibility of synthesis.

7. References

[1] K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, and J. Hirschberg, "ToBI: A standard for labeling english prosody," in *Proceedings of 2nd International Conference on Spoken Language Processing*, Banff, Alberta, Canada, October 1992, pp. 867–870.

[2] A. W. Black and P. Taylor, "The festival speech synthesis system: system documentation," Human Communication Research Centre, University of Edinburgh, Tech. Rep., January 1997. [Online]. Available: http://www.cstr.ed.ac.uk/projects/festival

[3] P. Taylor and A. W. Black, "Assigning phrase breaks from part-of-speech sequences," *Computer Speech and Language*, vol. 12, pp. 99–117, 1998.

[4] P. Roach, G. Knowles, T. Varadi, and S. Arnfield, "MARSEC: A machine-readable spoken english corpus," *Journal of the International Phonetic Association*, vol. 23, no. 1, pp. 47–53, 1993.

[5] A. Parlikar and A. W. Black, "A grammar based approach to style specific phrase prediction," in *Proceedings of Interspeech*, Florence, Italy, August 2011, pp. 2149–2152.

[6] C. J. van Rijsbergen, *Information Retrieval*. Butterworth, 1979.

[7] J. Bachenko, E. Fitzpatrick, and C. Wright, "A computational grammar of discourse-neutral prosodic phrasing in english," *Computational Linguistics*, vol. 16, pp. 155–170, Sep. 1990. [Online]. Available: http://dl.acm.org/citation.cfm?id=98377.98380

[8] M. Q. Wang and J. Hirschberg, "Automatic classification of intonational phrase boundaries," *Computer Speech and Language*, vol. 6, pp. 175–196, 1992.

[9] P. Koehn, S. Abney, J. Hirschberg, and M. Collins, "Improving intonational phrasing with syntactic information," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Istanbul, Turkey, June 2000.

[10] C. S. Fordyce and M. Ostendorf, "Prosody prediction for speech synthesis using transformational rule-based learning," in *Proceedings of the 5th International Conference on Spoken Language Processing*, Sydney, Australia, December 1998.

[11] P. A. Barbosa and G. Bailly, *Progress in speech synthesis*. New York: Springer Verlag, 1997, ch. Generation of pauses within the z-score model, pp. 365–381.

[12] H. Schmid and M. Atterer, "New statistical methods for phrase break prediction," in *Proceedings of the 20th international conference on Computational Linguistics*, Geneva, Switzerland, August 2004, pp. 659–665.

[13] N. Obin, P. Lanchantin, A. Lacheret, and X. Rodet, "Reformulating prosodic break model into segmental hmms and information fusion," in *Proceedings of Interspeech*, Florence, Italy, August 2011, pp. 1829–1832.

[14] E. Marsi, M. Reynaert, A. van den Bosch, W. Daelemans, and V. Hoste, "Learning to predict pitch accents and prosodic boundaries in dutch," in *Proceedings of Association for Computational Linguistics*, July 2003, pp. 489–496. [Online]. Available: http://dx.doi.org/10.3115/1075096.1075158

[15] M. Maragoudakis, P. Zervas, N. Fakotakis, and G. Kokkinakis, "A data-driven framework for intonational phrase break prediction," in *Text, Speech and Dialogue*, ser. Lecture Notes in Computer Science, V. Matoušek and P. Mautner, Eds. Springer Berlin / Heidelberg, 2003, vol. 2807, pp. 189–197.

[16] F. Liu, H. Jia, and J. Tao, "A maximum entropy based hierarchical model for automatic prosodic boundary labeling in mandarin," in *Proceedings of the 6th International Symposium on Chinese Spoken Language Processing*, Kunming, China, December 2008, pp. 1 –4.

[17] Z. Ying and X. Shi, "An RNN-based algorithm to detect prosodic phrase for chinese TTS," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, Salt Lake City, UT, USA, May 2001, pp. 809–812.

[18] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, 2003, pp. 160–167.

[19] D. J. Wales and J. P. K. Doye, "Global optimization by basin-hopping and the lowest energy structures of lennard-jones clusters containing up to 110 atoms," *The Journal of Physical Chemistry A*, vol. 101, pp. 5111–5116, 1997.

[20] A. W. Black, "CLUSTERGEN: A statistical parametric synthesizer using trajectory modeling," in *Proceedings of Interspeech*, Pittsburgh, Pennsylvania, September 2006, pp. 194–197.

[21] M. Ostendorf, P. J. Price, and S. Shattuck-Hufnagel, "The boston university radio news corpus," Boston University, Tech. Rep., March 1995. [Online]. Available: http://ssli.ee.washington.edu/papers/radionews-tech.ps

[22] J. R. Phillips. Online curve and surface fitting. [Online]. Available: http://www.zunzun.com

[23] L. M. Tomokiyo, K. Peterson, A. W. Black, and K. A. Lenzo, "Intelligibility of machine translation output in speech synthesis," in *Proceedings of Interspeech*, Pittsburgh, September 2006.

[24] A. Parlikar, A. W. Black, and S. Vogel, "Improving speech synthesis of machine translation output," in *Proceedings of Interspeech*, Makuhari, Japan, September 2010, pp. 194–197.