ENTROPY BASED CLASSIFIER COMBINATION FOR SENTENCE SEGMENTATION

M. Magimai.-Doss[1], *D. Hakkani-Tür*[1], *O. Çetin*[1], *E. Shriberg*[1,2], *J. Fung*[1], *N. Mirghafori*[1]

International Computer Science Institute, Berkeley, CA, USA[1]
SRI International, Menlo Park, CA, USA[2]

ABSTRACT

We describe recent extensions to our previous work on sentence segmentation. We extend the set of classification methods to exploit lexical, syntactic, speaker change, and prosodic features for this task with support vector machines. We propose a new *dynamic* classifier combination method for sentence segmentation, which is based on the entropy of each classifier and estimates a different weight for each example. Furthermore, we use the classifier combination with hidden event language models. The entropy-based classifier combination yields a 1% absolute improvement in F-Measure on Mandarin broadcast news when combined with the hidden event language model. In terms of NIST error rate the voting approach was better and resulted in a statistically significant 2.7% absolute reduction.

Index Terms— sentence segmentation, classifier combination, entropy, lexical and prosodic features, hidden event language model

1. INTRODUCTION

Sentence segmentation aims to enrich the unstructured word sequence output of automatic speech recognition (ASR) systems with sentence boundaries, to ease the further processing by both humans and machines. For instance, the enriched output of ASR (i.e., with sentence boundaries marked) can be used for later processing such as machine translation, question answering, or story/topic segmentation.

In the literature, the task of detecting sentence boundaries is seen as a two-class classification problem, and different classifiers have been investigated. [1] and [2] use a method that combines hidden Markov models (HMM) with N-gram language models containing words and sentence boundary tags associated with them [3]. This method was later extended with confusion networks in [4]. [5] provides an overview of different classification algorithms (boosting, hidden-event language models, maximum entropy models and decision trees) applied to this task for multilingual broadcast news. Besides the type of classifier, the use of different features has been widely studied. [2, 6] showed how prosodic features can benefit the sentence segmentation task. Investigations on prosodic and lexical features in the context of telephone conversations and broadcast news speech are also presented in [6, 5]. More recently, the use of syntactic features has been studied in [7].

In this paper, we extend our previous work on sentence segmentation for broadcast news [5] in the framework of the DARPA GALE program, where different classifiers, namely, decision trees, boosting, and maximum entropy models, were investigated. It was found that boosting was the best individual classifier. Furthermore, it was shown that performance can be further improved by combining the individual classifier probabilities with the probabilities estimated from hidden event language model (HELM).

While our previous work focused on individual classifiers and their combination with HELM, this paper focuses on combining the individual classifier outputs to improve the performance of sentence segmentation. More specifically, we study the use of the entropy-based classifier combination approach, which has been successfully used for ASR [8]. The entropy-based approach *dynamically* estimates the weight for each classifier based on its instantaneous output probabilities for each example and combines the output probabilities of the different classifiers before making the final decision. We compare this approach to a standard classifier decision combination approach, voting, and other static-weight-based classifier combination techniques such as linear regression and logistic regression. The predicted advantage of the entropy-based approach over linear regression and logistic regression is that it can generalize well to different data sets. Our results on the Mandarin TDT4 broadcast news database validate this prediction. The classifiers used in our studies are boosting, maximum entropy models, and support vector machines (SVM). Finally, we also study the combination of individual classifier output probabilities as well as the combined probabilities resulting from the entropy-based approach with HELM probabilities. The newly proposed *dynamic* classifier combination method for sentence segmentation further improves the performance. When we combine the outcome of classifier combination with HELMs, we obtain the best performance of 55.2% NIST error rate, a 2.7% absolute reduction from the NIST error rate of 57.9% obtained with the previous approach. With this new method, the F-measure also improves from 70.6% to 71.6%.

The rest of the paper is organized as follows. Section 2 formulates the sentence segmentation problem as a classification task, and then describes the different classifiers and classifier combination techniques that have been investigated. Sections 3 describes our experimental setup and the results are presented in Section 4. Finally, in Section 5 we conclude.

2. SENTENCE SEGMENTATION

Sentence segmentation can be considered as a binary boundary classification problem with "sentence-boundary" (s) and "non-sentence-boundary" (n) as classes [5]. For a given word sequence $\{w_1, ..., w_N\}$, the goal is to estimate the classes for boundaries $\{s_1, ..., s_N\}$, where $s_i,\ i = 1, ..., N$ is the boundary between w_i and w_{i+1}. Usually, this is done by training a classifier to estimate the posterior probability $P(s_i = k|o_i)$, where $k \in \{s, n\}$ and o_i are the feature observations for the word boundary s_i.

Ideally, the decision of the classifier is the class with maximum probability $P(s_i = k|o_i)$. However, in a sentence segmentation task the probability for sentence boundary $P(s_i = s|o_i)$ is compared against a threshold. If above the threshold, a decision of sentence boundary is made, else the boundary is marked as a non-sentence-boundary. As it will be seen later, for different classifiers and differ-

ent evaluation metrics the optimal threshold is different.

2.1. Classifiers

In this work, we have used boosting, maximum entropy and support vector machine (SVM) classifiers to estimate $P(s_i|o_i)$.

Boosting is an iterative learning algorithm that aims to combine "weak" base classifiers to come up with a "strong" classifier. At each iteration, a weak classifier is learned so as to minimize the training error, and a different distribution or weighting over the training examples is used to give more emphasis to examples that are often misclassified by the preceding weak classifiers. For this approach we use the *BoosTexter* tool described in [9] which has the advantage of discriminative training. Moreover, it can deal with large set of features both discrete and continuous valued, and has the capability to handle missing feature value.

Maximum entropy (MaxEnt) models are prominently used for natural language processing [10]. The main advantages of MaxEnt models are discriminative training, capability to handle a large set of features, flexibility in handling missing feature values, and convergence to a unique defined global optimum. In standard MaxEnt models, the features are discrete valued. However, the feature set used in our studies contains continuous valued features in addition to discrete valued features. The process of discretizing the continuous valued features for MaxEnt models is subject to research. In this work, we discretize the continuous valued feature by binning them to 10 classes similar to [5]. We use the open NLP toolkit to train the MaxEnt models [11].

SVMs are used in wide range of pattern recognition applications [12]. SVMs provide advantages similar to MaxEnt models in terms of discriminative training and capability to handle large set of features. In addition to it, SVMs can handle both discrete valued and continuous valued features. In our studies, we normalize the continuous valued features in the training data so as to have a zero mean and unit variance. In our data, there are instants where a feature value may be missing (for the words that ASR outputs as reject labels). In such a case, we introduce a new token for unknown discrete valued features. For continuous valued features, we use the mean value, that is 0.0. When training SVM classifier we found that the choice of the kernel type is not obvious. When using mainly lexical features (discrete valued) the best performance is achieved with a linear kernel, whereas while using both lexical and prosodic features (both discrete and continuous valued) the use of a polynomial kernel of degree 2 yields the best performance. For our studies, we use the SVMlight toolkit [13].

2.2. Classifier Combination

Combining classifiers is a well researched topic in machine learning and spoken language processing [14, 15, among others]. Some of the most common classifier combination methods used in the literature include voting, and linear and logistic regression. The general objective of classifier combination is to exploit the complementary information between the classifiers. In a sense, the different classifiers in a classifier combination can be seen as a collection of weak classifiers, where each classifier can solve some different difficult problems. Then, the process of combination involves combining the decisions of classifiers or assigning a weight to each classifier's output evidence (in our case $P(s_i|o_i)$) and combining the evidence so as to reduce the objective error. The weights can be estimated statically, that is, *a priori*, on held-out data or development data, for example linear regression or dynamically, for example inverse entropy combination. In this paper, we have investigated the both.

Let $c \in \{1, ..., C\}$ and $P_c(s_i = k|o_i)$ denote the different classifiers (in our case $C = 3$) and their output probabilities at instant i. Given these, we investigate the following classifier combination techniques:

- Inverse entropy
- Voting
- Linear regression
- Logistic regression

2.2.1. Inverse Entropy

Given the instantaneous classifier output probabilities, the idea of inverse entropy combination is to assign large weights to classifiers that are more confident in their decisions and small weights to classifiers that are less confident in their decisions [8]. The confidence is measured in terms of entropy ent_c of classifier output probabilities, that is,

$$ent_c = \sum_{k=1}^{K} -P_c(s_i = k|o_i) \cdot \log_2(P_c(s_i = k|o_i)), \quad \forall c \quad (1)$$

where K is the number of output classes, which in our case is 2.

The weight ω_c for each classifier is then estimated as

$$\omega_c = \frac{\frac{1}{ent_c}}{\sum_{c=1}^{C} \frac{1}{ent_c}}, \quad \forall c \quad (2)$$

Having estimated the weight for each classifier, the output probabilities of the classifiers can be combined in two different ways:

1. Sum rule

$$P(s_i = k|o_i) = \sum_{c=1}^{C} \omega_c \cdot P_c(s_i = k|o_i), \quad \forall k \quad (3)$$

2. Product rule

$$P(s_i = k|o_i) = \frac{1}{Z} \cdot \prod_{c=1}^{C} P_c(s_i = k|o_i)^{\omega_c}, \quad \forall k \quad (4)$$

where Z is a normalization factor.

The decision about output class is then made based upon $P(s_i = s|o_i)$.

2.2.2. Voting

The voting technique looks for agreement between classifiers output decisions. Typically, each classifier votes for each output class based on its decision. The final decision or the output class is the class getting the maximum number of votes.

In our case, all classifiers have one vote and each classifier c decides its vote by comparing $P_c(s_i = s|o_i)$ to the threshold.

2.2.3. Linear Regression

In a linear regression classifier combination, the combined probability for the class sentence boundary ($P(s_i = s|o_i)$) is estimated as the linearly weighted sum of classifier output probabilities for sentence boundary class $P_c(s_i = s|o_i)$:

$$P(s_i = s|o_i) = a + \sum_{c=1}^{C} \omega_c \cdot P_c(s_i = s|o_i) \quad (5)$$

where a is a constant. The constant a and the weights w_c are optimized on development data, and are frozen for the test data. In our studies, we use the MATLAB function *regress* to estimate the weights and constant.

2.2.4. Logistic Regression

The classifier output probabilities can be combined using logistic regression by treating them as predictor variables of the logistic function and, estimating the combined probability for the sentence boundary class ($P(s_i = s|o_i)$) as

$$P(s_i = s|o_i) = \frac{1}{e^{-(a+\sum_{c=1}^{C} \omega_c \cdot P_c(s_i=s|o_i))}} \quad (6)$$

where the constant a and weights w_c for each classifier c are estimated on a development data similar to linear regression. In our work, we estimate the optimal a and w_c on the development data using the Newton-Raphson method [16].

3. EXPERIMENTAL SETUP

3.1. Data

To evaluate our approach, we have used a subset of the TDT-4 Mandarin Broadcast News corpus. Table 3.1 lists the properties of the training, development and test sets, which are picked out in a time order (that is, all shows in the training set precede the development and test set shows). The sentence segmentation experiments have been performed on the ASR output that was used in our earlier work [5].

	Training	**Dev**	**Test**
No. of shows	131	17	17
No. of sentence units	19,643	2,903	2,991
No. of examples	481,419	72,152	77,915

Table 1. *Description of the Training, Development (Dev) and Test data set.*

3.2. Feature sets

To study the use of prosodic features, we use two feature sets, namely, *WP* and *ALL*:

- *WP*: includes only the words around the boundary, pause duration, and their N-grams
- *ALL*: includes *WP*, prosodic features, speaker turn, and part-of-speech (POS) tag N-grams
  In addition to the word-based prosodic features related to pitch, pitch slope, energy and pause duration between words that were used in our earlier work [5], we also use speaker normalized versions of pitch and energy features. We used the ICSI-SRI speaker diarization system to divide the audio data into hypothetical speakers [17], and pitch baseline and pitch ranges were estimated on each speaker, which were then used as parameters in the normalization. The speaker turn features were extracted from the diarization segmentation. We use the POS tag features that were used in [5].

3.3. Evaluation Metrics

We evaluate our systems in terms of F-measure (FM) and NIST error ($NIST$) [18]. If I is the number of insertions (i.e., false positives), D is the number of deletions (i.e., false negative) and C is the number of sentence boundaries correctly recognized, then:

$$precision = \frac{C}{(C+I)}, \quad recall = \frac{C}{(C+D)}$$

$$FM = \frac{2.precision.recall}{(precision + recall)} \quad (7)$$

$$NIST = \frac{(I+D)}{(C+D)} \quad (8)$$

Here note that unlike FM, $NIST$ does not have an upper bound.

4. RESULTS

Tables 2 and 3 show sentence segmentation performance with individual classifiers and combination classifiers with various methods for the development and test sets, respectively, using both the *WP* and *ALL* feature sets. In all the tests, we use the optimum thresholds on the development set for computing the NIST error and F-measure on the test set. With the *ALL* features, out of the three classifiers, SVM performs the best with 57.6% NIST error rate and 69.8% F-measure. The inverse entropy (IE) classifier combination method results in the best F-measure of 70.9% on the test set, and the voting method results in the lowest NIST error rate of 56.6%. The performance difference between the voting and IE methods and the linear and logistic regression methods on the test set is statistically significant [1] the first two performing better. The MaxEnt classifier performs the best on WP feature set, but performs poorer compared to boosting and SVM on feature set ALL. This may be primarily due to the way the discretization of the continuous valued feature is performed.

	WP		*ALL*	
	FM	$NIST$	FM	$NIST$
Boosting	68.0	64.9	72.6	54.3
MaxEnt	**68.5**	**64.5**	69.1	60.3
SVM	67.9	64.9	**72.7**	**52.2**
IE (sum)	69.4	61.8	73.6	51.8
IE (prod)	69.2	61.7	73.5	51.8
Voting	69.0	62.4	73.4	**51.6**
Linear	69.1	61.9	73.6	52.2
Logistic	**69.6**	**61.4**	**73.9**	51.9

Table 2. *Sentence segmentation performance on the development set for feature sets WP and ALL. The FM an $NIST$ are expressed in %. IE denotes inverse entropy, and sum and prod refer to the sum and product combination rules in* (3) *and* (4)*, respectively. Linear denotes the linear regression technique, and Logistic denotes the logistic regression technique. Note that the linear regression and logistic regression weights were estimated on the development set. The best performance for individual classifiers and classifier combination is in boldface.*

	WP		*ALL*	
	FM	$NIST$	FM	$NIST$
Boosting	64.4	71.5	68.9	61.3
MaxEnt	**65.7**	**70.8**	67.3	63.5
SVM	64.7	71.9	**69.8**	**57.6**
IE (sum)	**66.5**	**68.2**	70.7	57.6
IE (prod)	66.1	68.6	**70.9**	57.3
Voting	66.4	68.7	70.6	**56.6**
Linear	64.1	70.2	67.4	60.0
Logistic	64.1	69.7	68.6	59.6

Table 3. *Sentence segmentation performance on the test set for feature sets WP and ALL. The FM an $NIST$ are expressed in %. IE denotes inverse entropy, and sum and prod refer to the sum and product combination rules in* (3) *and* (4)*, respectively. Linear denotes the linear regression technique, and Logistic denotes the logistic regression technique. The linear regression and logistic regression weights are estimated on the development set. The best performance for individual classifiers and classifier combination is in boldface.*

[1] according to the Z-test with 95% confidence interval

Table 4 shows the performance when we convert the posterior probabilities from each classifier and combination classifier and use them as state observation likelihoods in the hidden event language model (HELM). The HELM used for this experiment is a 4-gram model and is trained using all the text in the training set, as well as other data. We achieve the best NIST error of 55.2% on the test set when the combined probabilities are estimated as class probabilities averaged across classifiers.

	Dev		Test	
	FM	$NIST$	FM	$NIST$
Boosting+HELM	74.1	50.9	70.6	57.9
MaxEnt+HELM	71.5	55.0	69.2	61.1
SVM+HELM	**75.5**	**48.9**	**71.9**	**56.7**
IE (sum)+HELM	74.3	49.5	71.3	56.3
IE (prod)+HELM	**74.5**	49.5	**71.6**	56.1
Avg. Prob+HELM	74.2	49.5	70.8	**55.2**

Table 4. *Results of sentence segmentation studies with HELM for feature set ALL on the development (Dev) and test set. The FM an $NIST$ are expressed in %. IE(sum): the combined classifier output probabilities are estimated using* (3)*, IE(prod): the combined classifier output probabilities are estimated using* (4)*, Avg. Prob: Class probabilities averaged across classifiers. The best performance for individual classifiers and classifier combination is in boldface.*

5. CONCLUSIONS

We have described the recent extensions to our previous work on sentence segmentation, where we extend the set of classification methods with SVMs and combine classification methods with a new, dynamic, entropy-based classifier combination method, which was already shown to be useful for ASR. We show improvements when we use an extended set of prosodic features in addition to pause duration with all classifiers. Furthermore, to model the sequence information, we incorporate the combined classifier output with hidden event language models. We show statistically significant improvements for both NIST error rate and F-measure.

Acknowledgments
This work is supported by DARPA under Contract No. HR0011-06-C-0023. The views are those of the authors and do not necessarily represent the views of the funding agency. The authors thank Yang Liu, Matthias Zimmermann, Luke Gottlieb and Gokhan Tür.

6. REFERENCES

[1] Y. Gotoh and S. Renals, "Sentence boundary detection in broadcast speech transcripts," in *Proc. ISCA ITRW Workshop*, Paris, 2000.

[2] E. Shriberg, A. Stolcke, D. Hakkani, and G. Tur, "Prosody-based automatic segmentation of speech into sentences and topics," *Speech Communication*, vol. 31, pp. 127–154, 2000.

[3] A. Stolcke and E. Shriberg, "Automatic linguistic segmentation of conversational speech," in *ICSLP*, 1996.

[4] D. Hillard, M. Ostendorf, A. Stolcke, Y. Liu, and E. Shriberg, "Improving automatic sentence boundary detection with confusion networks," in *Proc. HLT-NAACL*, Boston, MA, 2004.

[5] M. Zimmermann, D. Hakkani-Tür, J. Fung, N. Mirghafori, E. Shriberg, and Y. Liu, "The ICSI+ multi-lingual sentence segmentation system," in *Proc. ICSLP*, Pittsburgh, PA, 2006.

[6] Y. Liu, E. Shriberg, A. Stolcke, B. Peskin, J. Ang, D. Hillard, M. Ostendorf, M. Tomalin, P. Woodland, and M. Harper, "Structural metadata research in the EARS program," in *Proc. ICASSP*, 2005.

[7] B. Roark, Y. Liu, M. Harper, R. Stewart, M. Lease, M. Snover, I Shafran, B. Dorr, J. Hale, A. Krasnyanskaya, and L. Yung, "Reranking for sentence boundary detection in conversational speech," in *ICASSP*, 2006.

[8] H. Misra, H. Bourlard, and V. Tyagi, "New entropy based combination rules in HMM/ANN multi-stream ASR," in *ICASSP*, 2003.

[9] R. E. Schapire and Y. Singer, "BoosTexter: A boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2-3, pp. 135–168, 2000.

[10] A. Berger, V. J. Della Pietra, and S. A. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 30–71, 1996.

[11] A. Ratnaparkhi, *Maximum entropy models for natural language ambiguity resolution*, Ph.D. thesis, University of Pennsylvania, 1998, http://maxent.sourceforge.net/.

[12] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1995.

[13] T. Joachims, "Making large-scale svm learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schlkopf, C. Burges, and A. Smola, Eds. MIT Press, 1999, http://svmlight.joachims.org/.

[14] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 226–239, 1998.

[15] M. Karahan, D. Hakkani-Tür, G. Riccardi, and G. Tur, "Combining classifiers for spoken language understanding," in *ASRU*, 2003.

[16] A. Agresti, *Categorical Data Analysis*, John Wiley and Sons, 1990.

[17] C. Wooters, J. Fung, B. Peskin, and X. Anguera, "Towards robust speaker segmentation: the ICSI-SRI fall 2004 diarization system," in *RT-04F Workshop*, 2004.

[18] "The rich transcription fall 2003," http://nist.gov/speech/tests/rt/rt2003/fall/docs/rt03-fall-eval-plan-v9.pdf.