MULTI-STREAM LANGUAGE IDENTIFICATION USING DATA-DRIVEN DEPENDENCY SELECTION

Sonia Parandekar, Katrin Kirchhoff

{sonia,katrin}@ee.washington.edu
Department of Electrical Engineering
University of Washington, Seattle, USA

ABSTRACT

The most widespread approach to automatic language identification in the past has been the statistical modeling of phone sequences extracted from speech signals. Recently, we have developed an alternative approach to LID based on n-gram modeling of parallel streams of articulatory features, which was shown to have advantages over phone-based systems on short test signals whereas the latter achieved a higher accuracy on longer signals. Additionally, phone and feature streams can be combined to achieve maximum performance. Within this “multi-stream” framework two types of statistical dependencies need to be modeled: (a) dependencies between symbols in individual streams and (b) dependencies between symbols in different streams. The space of possible dependencies is typically too large to be searched exhaustively. In this paper, we explore the use of genetic algorithms as a method for data-driven dependency selection. The result is a general framework for the discovery and modeling of dependencies between multiple information sources expressed as sequences of symbols, which has implications for other fields beyond language identification, such as speaker identification or language modeling.

1. INTRODUCTION: N-GRAM MODEL APPROACHES TO LANGUAGE IDENTIFICATION

Automatic language identification (LID) continues to be of considerable importance for multilingual speech applications. Several approaches to LID have been developed in the past which make use of acoustic, prosodic, phonetic-phonotactic or lexical information. Of these, the phonotactic approach (e.g.[1]) has emerged as the most widespread and flexible technique. This approach assumes that language-discriminating information is encoded in the statistical regularities of phone sequences in different languages. As a first step, the speech signal is mapped to a sequence of phone symbols, $\Phi = \phi_1, \phi_2, ..., \phi_N$, using acoustic models such as Hidden Markov Models (HMMs). Statistical n-gram models are then trained on the resulting phone labels. An n-gram model specifies a set of probability distributions of a phone given a context of $n-1$ and the language L:

$$P(\phi_1, \phi_2 ..., \phi_N|L) = \prod_{i=n}^{N} P(\phi_i|\phi_{i-1}, ..., \phi_{i-n+1}, L) \tag{1}$$

During language identification, the phone sequence derived from the test speech signal is scored against each of the language-specific n-gram models. The language of the n-gram model for which the highest score is obtained is then hypothesized as the true language (L^*):

$$L^* = argmax_L P(\phi_1, \phi_2, ..., \phi_N|L) \tag{2}$$

In our recently developed feature-based approach [2, 3], articulatory feature sequences are used in place of phone sequences. These features characterize different articulatory properties of the speech signal and are arranged into five separate groups (*manner* of articulation, *consonantal place* of articulation, *vowel place* of articulation, *front-back* tongue position and lip *rounding*) and Acoustic models are built for each feature value, analogous to acoustic phone models. Using these models, parallel streams of feature sequences, one for each feature group, are derived from a given speech signal. Language-specific n-gram models are trained for each feature stream. Analogous to the n-gram probability of a phone sequence, the probability of a feature sequence in a particular feature stream $\mathcal{F} = f_1, ...f_N$ given a language L, $P(\mathcal{F}|L)$, is defined as:

$$P(\mathcal{F}|L) = \prod_{i=n}^{N} P(f_i|f_{i-1}, ..., f_{i-n+1}, L) \tag{3}$$

The probability of an ensemble of K feature streams $\mathcal{F}_1, ..., \mathcal{F}_K$ given language L, $P(\mathcal{F}_1, ..., \mathcal{F}_K|L)$, is defined as:

$$P(\mathcal{F}_1, ..., \mathcal{F}_K|L) = \mathcal{C}(P(\mathcal{F}_1|L), , ..., P(\mathcal{F}_K|L)) \tag{4}$$

where $\mathcal{C}$ is some combination function, e.g. the product rule

$$P(\mathcal{F}_1, ..., \mathcal{F}_K|L) = \prod_{k=1}^{K} P(\mathcal{F}_k|L) \tag{5}$$

In our previous work we showed that both feature-based and phone-based approaches achieved comparable performance overall but that the feature-based system obtained a significantly higher performance on very short test signals ($\leq$ 3sec.) whereas the phone-based system achieved a higher accuracy on longer test signals. Due to the complementary nature of the two approaches, they can be combined to achieve maximum performance. A seamless way of integrating the phone and feature-based systems is to treat the stream of phones as an additional stream within the set of articulatory feature streams. The equation for language classification would now become

$$L^* = argmax_L P(\mathcal{F}_1, ..., \mathcal{F}_K, \Phi|L) \tag{6}$$

Naturally, this can be extended to using multiple phone sequences, as is the standard in some phone-based LID systems. In all “multi-stream” models of this type, two sets of dependencies need to be modeled: (a) dependencies within individual streams, and (b) dependencies across different streams. The model in Equation 5 assumes that all streams are independent given the language and thus ignores dependencies of type (b), which is clearly an oversimplification. The main objective of this study is to explore how statistical dependencies between different information streams can be detected and modeled more adequately.

2. MODELING CROSS-STREAM DEPENDENCIES

In our work, the phone stream is treated as an additional $(K+1)^{th}$ feature stream. The baseline phone and feature systems described above model the probabilities of a symbol at a given time t conditioned on symbols at previous time positions within the same stream. Possible dependencies on variables in other streams are not taken into account. However, the different streams are extracted from the same speech signal and conditioning a symbols on variables in other streams might therefore yield additional gains. A cross-stream model can be represented more formally as:

$$P(f_i^j) = P(f_i^j | f_{i-1}^j, f_{i-2}^j, ..., f_{i-n+1}^j, \mathcal{F} \setminus \{j\}) \qquad (7)$$

where, $\mathcal{F} \setminus \{j\}$ represents some subset of the set of all features minus those in stream j. In general, for a fixed context length of n and K streams, there are $n * K - 1$ conditioning features, viz. the $n-1$ previous features in the same stream and all n features in the current context window in the $K-1$ remaining streams. To find the optimal combination of some or all of these conditioning variables, we in principle need to conduct an exhaustive search over all possible subsets. The number of possible subsets, $\sum_{i=1}^{j} \binom{j}{i}$, where $j = n*K-1$, is prohibitively large and cannot be searched exhaustively.

For models with cross-stream dependencies, the conditioning and the dependent streams need to be aligned to determine joint frequencies. A simultaneous alignment of all streams at the frame level typically leads to multiple repetitions of the same symbol within a stream and across sub-groups of streams. We noticed in previous experiments that such repetitions decrease accuracy significantly as they tend to dominate the n-gram scores. To overcome this problem, we separately align the sets of conditioning and dependent variables corresponding to different cross-stream dependencies and, within each set, only use those vectors of variable values for scoring where at least one value changes. The scores for the different stream groupings, normalized by the number of vectors considered, are then combined using Equation 5. Weighted combination using another classifier, e.g. a multi-layer perceptron, was not shown to yield any advantages beyond product combination in the past.

In our previous work [2] pair-wise cross-stream dependencies in a purely feature-based system were identified using a greedy search technique. Their integration yielded improvements in LID accuracy; however, only a small subset of all possible dependencies was explored. In this study we use a more powerful search technique, viz. Genetic Algorithms (GA), as described in the next section.

3. GENETIC ALGORITHMS FOR DEPENDENCY SELECTION

Genetic Algorithms are a general search and optimization technique inspired by natural evolutionary processes. Some of the distinguishing characteristics of GAs are the following:

1. GAs do not deal directly with problem parameters themselves but with binary string encodings of individual problem solutions. Different solutions are created by applying genetic operators modifying these strings, as described below. Since these operators are applied probabilistically, the search is guided towards various unexplored parts of the search space that might potentially contain better solutions. In contrast, deterministic techniques always search within a fixed pre-defined area of the search space.
2. GAs work with a *population* of potential solutions i.e. sets of parameter values (in the form of encoded strings) rather than a single solution. GAs thus simultaneously explore several sections of the search space for a particular problem, which often prevents them from converging on a local optimum.
3. GAs employ a user defined *fitness function* to determine the goodness of each solution or the entire population. At each iteration of the genetic search, the current population is evaluated and modified to move to a higher fitness value.

GA operators are simple, well defined functions that are applied to members within its population. The most basic operators employed by GAs are *reproduction* or *selection*, *crossover*, and *mutation*. Reproduction is a process by which individual strings are copied into a pool from which strings for the next iteration are selected. Strings with a higher fitness value have a greater probability of contributing to this pool. During crossover, pairs of members from the pool are selected and new strings are produced by swapping characters from the original pair at randomly selected positions. Mutation is the occasional random alteration of value of a bit or digit in the string. Reproduction and crossover try to preserve good partial solutions from one iteration to the next. Mutation is of secondary importance compared to reproduction and crossover and is used mainly to maintain the heterogeneity of the population and to prevent premature convergence. GA search involves the following steps:

- encode the problem parameters (i.e decision variables) as binary strings;
- determine an appropriate fitness function;
- randomly generate an initial population of strings;
- while the termination criterion has not been reached
  - evaluate the fitness of each individual in the population and select a pool of solutions for the next generation
  - apply crossover and mutation

The operators work on successive generations of solutions, with each generation producing more and more refined solutions. The algorithm stops when some termination criterion (usually a specific value of the fitness function or value of its change over several generations) is satisfied. Several alternatives are available for the specific implementation of the GA operators. In our work we investigated different implementations, namely roulette wheel, stochastic universal sampling and tournament for selection, and one-point, two-point and uniform crossover operators. The choice of operator implementation had an impact on efficiency of the search but rarely on the final outcome and we settled upon tournament selection and uniform crossover for most of our experiments. The GA can be made more powerful by integrating advanced operators and techniques. One such technique that we have adopted is an *elitist* model, which ensures that the best individual of a generation is preserved in the next generation.

In order to apply GA-based search to the problem of dependency selection, the set of conditioning features for any given dependent feature variable needs to be encoded in a string. Consider a feature variable f_i^m, with a potential set of conditioning variables defined by:

$$< f_{i-2}^k, f_{i-1}^k, f_i^k, f_{i-2}^l, f_{i-1}^l, f_i^l, f_{i-2}^m, f_{i-1}^m >$$

where the context length of interest is 3. This conditioning set can be represented by an eight bit binary string with 1 representing presence and 0 representing absence of the dependency. For example, 10111001 would imply that f_i^m is conditioned on:

$$< f_{i-2}^k, f_i^k, f_{i-2}^l, f_{i-1}^l, f_{i-1}^m >.$$

It is important to prevent circular dependencies in order to obtain valid probability distributions. This occurs, for instance, when f_i^m is conditioned on f_i^k and vice versa. One method of overcoming this problem is to restrict the potential conditioning set for each dependent variable such that no circular dependencies are possible. The individual streams are arranged in an ordered tuple, for example: $< j, k, l, m >$. Any feature variable can be conditioned only on other feature variables in its own stream and those preceding it in the tuple. Though this effectively implies that the GA search will now be exploring a more restricted search space, different orderings of the set of features can be evaluated to get the optimal set of dependencies. Since the final goal is to improve on language identification accuracy, we use LID performance on a held-out development set as the fitness function.

4. CORPUS AND BASELINE SYSTEMS

Experiments reported in this paper are based on the OGI-TS corpus [4] of telephone speech data from 10 different languages (English, Farsi, French, German, Japanese, Korean, Mandarin, Spanish, Tamil and Vietnamese). We follow the same division of the corpus into training, development and evaluation sets as defined in [5], which is identical to the definitions included in the LDC distribution. These sets contain 4650, 1898 and 1848 utterances for training, development and evaluation respectively. Each of the three sets contains approximately the same number of utterances per language. It is important to note that unlike many previous studies which have excluded speech signals shorter than 10 seconds or only used a subset of the 10 languages, our results are based on a 10-way forced choice including all languages and signal files of all lengths.

The first step in our baseline LID system is speech/non-speech segmentation, which is performed by a neural network trained on a hand-labeled subset of the training data, followed by temporal smoothing of the network outputs. The speech segments are then converted to 12 mel-frequency cepstral coefficients, log energy, and first-order temporal derivatives, yielding 26-dimensional feature vectors. Based on this acoustic representation, Hidden Markov Models (HMMs) with 3 states and 2 Gaussian mixture components each are trained for 26 features grouped into the following five streams: manner of articulation (mann), consonantal place of articulation (cpl), vowel place of articulation (vpl), lip rounding (rd) and front-back position of the tongue (fb). Furthermore, 8 models are used for silence and various types of background noises. Individual noise models are trained for each feature group. The total number of models in the feature based system is 66. The trained acoustic models are then used to generate feature labels by unconstrained recognition. For each stream, trigram models with Witten-Bell smoothing are trained using the SRILM toolkit [6] with an extension module for multi-stream n-gram models implemented by Jeff Bilmes (U Washington). Explicit duration modeling was incorporated into the n-gram models by relabeling feature labels in accordance with their temporal duration: for each feature, a duration histogram was estimated and all labels above the 25th percentile of the distribution were relabeled as long features, others were relabeled as short features. This split, as well as the final selection of split labels to be incorporated into the n-gram modeling component, was optimized on the development set. This leads to a total of 89 feature models, arranged in sets of 19,21,21,15 and 13 models respectively, for the articulatory feature streams mentioned above. The phone system is trained on the same acoustic representation and contains 133 HMMs with 3 states and 4 Gaussian mixture components each. Phone trigrams are used.

5. EXPERIMENTS AND RESULTS

Table 1 shows the baseline error rates for the phone, feature and combined system. System combination was done as specified in Equation 5. In our previous work we concentrated on improving the feature-based approach by applying techniques such as explicit duration modeling for features, as explained in Section 4. These techniques were not applied to the phone-based system, which is why the baseline feature-based system shows a much better performance. However, due to the complementary nature of the two approaches, their combination still yields significant improvements in LID accuracy (significant at the 0.0002 and 0.002 level, respectively, using a difference of proportions significance test). It should be noted that the number of parameters required for the phone trigrams, i.e. 133^3 is much larger than the number of parameters required for the all feature trigrams combined, i.e. $19^3 + 21^3 + 21^3 + 15^3 + 13^3$.

Set	Phone	Feature	Combined
Dev. set	49.84	58.96	64.54
Eval. set	47.99	57.30	62.17
# params	2.35M	30K	2.38M

Table 1. LID accuracy (%) and number of n-gram parameters for phone-based, feature-based and combined LID systems.

Search	Dev. Set	Eval. Set	# params	dep.s
Group A	60.01	58.44	31K	1
Group B	61.85	60.06	33K	2
Group C	64.54	62.17	2.38M	none
Group D	54.21	51.35	2.35M	5

Table 2. LID accuracy (%), number of system parameters, and number of selected cross-stream dependencies .

Our next step was to incorporate explicit cross-stream dependencies, with the goal of not only obtaining the maximum LID accuracy but also of minimizing the number of parameters needed. The entire set of possible dependencies included within-stream feature and phone dependencies (i.e. standard n-grams), cross-stream dependencies among feature streams, as well as cross-stream dependencies between individual feature streams and the phone stream. N-gram orders of up to $n = 3$ were considered. We applied GA-based search to the following subsets of dependency models:

- within-stream + cross-stream dependencies for features only (Group A), i.e. standard feature n-grams plus allowing conditioning features in streams other than the current one;
- as Group A, but additionally allowing phones to be conditioned on features or vice versa (Group B);
- as Group B, but additionally including standard phone n-grams (Group C);
- standard phone n-gram plus allowing phones to be conditioned on features or vice versa; no feature n-grams (Group D).

Results are shown in Table 2. We made the following observations: first, the accuracy on the development set always increases significantly, which is not surprising since the fitness function was directly designed to maximize accuracy on this set. On the evaluation set, significant gains (p = 0.05) were obtained for Groups B and D, compared to the baseline systems. Second, cross-stream dependencies were selected for Groups A and B, which do not include phone n-grams among the set of possible models, but not in C, which does include phone trigrams. It seems that phone n-grams and feature n-grams plus cross-stream dependencies are two different ways of modeling similar information. These two approaches are associated with different accuracy-cost tradeoffs: whereas the inclusion of phone n-grams leads to the best accuracy overall it comes at a significant cost in terms of the number of

Search	Dev. Set	Eval. Set	# params	dep.s
Group A	58.48	58.98	31K	1
Group B	61.59	58.93	33K	1
Group C	64.54	62.17	2.38M	none
Group D	52.16	49.78	2.3M	1

Table 3. LID accuracy (%), number of system parameters, and number of selected cross-stream dependencies for greedy dependency selection.

Set	Phone	Feature	Combined
Dev. set	48.87	62.14	67.23
Eval. set	47.99	62.07	65.02

Table 4. LID accuracy (%) for phone-based, feature-based and combined LID systems using duration-specific n-grams.

parameters (see Table 2, column 3). The feature-based system, by contrast, has a lower accuracy but only a fraction of the parameters needed for the phone-based system. Finally, in the course of many experiments using GA-based search we noticed that it is impossible to predict which dependencies are selected under which conditions, indicating strong non-linear interactions between different conditioning variables. This emphasizes the advantages of using the Genetic Algorithm rather than heuristic or other conventional search techniques for dependency selection. As a comparison, we ran dependency selection experiments for the three groups using greedy search as described in [2]. The results (Table 3) show that GA-based search did indeed lead to better performance in most cases.

The OGI-TS corpus consists of spontaneous and non-spontaneous utterances ranging from a few seconds to a maximum of about one minute. Whereas short utterances include enumerations such as the days of the week, longer utterances are spontaneous narrations. Utterance duration is therefore related to speaking style and vocabulary effects. Better results might be obtained if both n-gram model training and GA search were applied to different duration-specific subsets of utterances separately. We sorted training utterances into different groups based on their length. Within these groups, the sets of development utterances were randomly sub-sampled to ensure that each language had roughly the same number of samples, which prevents language-specific biases in the GA search. For each category, within-stream and/or cross-stream dependency models trained on utterances in that category alone were considered for scoring if, on development data, they outperformed models trained on the entire training set. The results in Table 4 show that duration-specific n-grams are indeed beneficial for LID accuracy. Individual GA searches were then conducted separately for each durational category, using the same configuration as in Tables 2 and 3. Results from these experiments for different duration categories are shown in Table 5. We can see that there is hardly any additional advantage due to using dependency selection conditioned on utterance length - we even see a decrease in performance for Groups A and C. Since utterances are divided into sub-groups based on their lengths, there may not be enough data in the dev set of each group to obtain generalizable models and the Genetic Algorithm may be over-training on the development set.

6. SUMMARY AND CONCLUSIONS

In this paper we have demonstrated the importance of incorporating cross-stream dependencies into multi-stream models for LID.

	duration-based GA		standard GA	
Search	Dev. Set	Eval. Set	Dev. Set	Eval. Set
Group A	64.74	59.81	63.35	61.95
Group B	67.15	63.35	67.44	63.25
Group C	69.69	63.35	67.23	65.02
Group D	60.68	52.57	54.21	51.35

Table 5. LID accuracy (%) for duration-specific ngrams plus duration-specific GA search vs. standard GA search.

Our conclusions are that (a) due to the complementary nature of the two approaches, combining phone and feature-based information streams leads to significant improvements in LID accuracy; (b) further significant improvements in accuracy can be obtained by explicitly modeling dependencies across these different streams; (c) Genetic Algorithms outperform heuristic search for the purpose of selecting the best set of dependency models from the large space of all possible combinations. Applying GA-based search to different subclusters of utterances separately did not yield any significant improvement compared to GA search using all available data collectively. We also noticed the danger of over-training the GA to the development data, especially when cluster sizes are small. In the future, we intend to incorporate explicit penalization factors for model complexity into the fitness function. The approach presented here can be applied to a number of different scenarios, e.g. standard phonotactic LID systems that use multiple phone streams and to recently developed systems for speaker identification based on similar approaches [7, 8]. Furthermore, the framework is general enough to be able to accommodate any other information source which can be expressed as sequences of discrete symbols, such as sequences of HMM state indices or prosodic symbols.

Acknowledgements
The work presented in this paper was funded by US DoD.

7. REFERENCES

[1] M.A. Zissman, “Comparison of four approaches to automatic language identification of telephone speech,” *IEEE Trans. Speech and Audio Processing*, vol. 4(1), pp. 31–44, 1996.

[2] K. Kirchhoff and S. Parandekar, “Multi-stream statistical language modeling with application to automatic language identification,” in *Proceedings of Eurospeech-01*, 2001, pp. 803–806.

[3] K. Kirchhoff, S. Parandekar, and J. Bilmes, “Mixed-memory Markov models for automatic language identification,” in *Proceedings of ICASSP*, 2002, pp. 2841–2844.

[4] Y.K. Muthusamy et al., “The OGI multi-language telephone speech corpus,” in *Proceedings of ICSLP-92*, 1992.

[5] Y.K. Muthusamy, *A Segmental Approach to Automatic Language Identification*, Ph.D. thesis, Oregon Graduate Institute, 1993.

[6] A. Stolcke, “SRILM - an extensible language modeling toolkit,” in *Proceedings of ICSLP*, 2002.

[7] W. Andrews, M. Kohler, and J. Campbell, “Phonetic speaker recognition,” in *Proceedings of Eurospeech*, 2001, pp. 2517–2520.

[8] J. Qin, T. Schultz, and A. Waibel, “Speaker identification using multilingual phone string,” in *Proceedings of ICASSP*, 2002, pp. 145–148.