Finding Complex Features for Guest Language Fragment Recovery in Resource-Limited Code-Mixed Speech Recognition

Aaron Heidel, Hsiang-hung Lu and Lin-shan Lee, *Member, IEEE*

***Abstract*—The rise of mobile devices and online learning brings into sharp focus the importance of speech recognition not only for the many languages of the world but also for code-mixed speech, especially where English is the second language. The recognition of code-mixed speech, where the speaker mixes languages within a single utterance, is a challenge for both computers and humans, not least because of the limited training data. We conduct research on a Mandarin-English code-mixed lecture corpus, where Mandarin is the host language and English the guest language, and attempt to find complex features for the recovery of English segments that were misrecognized in the initial recognition pass. We propose a multi-level framework wherein both low-level and high-level cues are jointly considered; we use phonotactic, prosodic, and linguistic cues in addition to acoustic-phonetic cues to discriminate at the frame level between English- and Chinese-language segments. We develop a simple and exact method for CRF feature induction, and improved methods for using cascaded features derived from the training corpus. By additionally tuning the data imbalance ratio between English and Chinese, we demonstrate highly significant improvements over previous work in the recovery of English-language segments, and demonstrate performance superior to DNN-based methods. We demonstrate considerable performance improvements not only with the traditional GMM-HMM recognition paradigm but also with a state-of-the-art hybrid CD-HMM-DNN recognition framework.**

***Index Terms*—speech recognition, bilingual, code-mixing, language identification.**

I. Introduction

CODE mixing (CM) occurs when a speaker mixes languages within a single utterance [1]; some know this as code switching [2]. For example,

Given 前面這些 state 的 history
(Given the history of these states)

你如果修過相關的 sampling theory 的課的話
(If you have taken classes on sampling theory before)

我就是把這個 D sub I 放到這個 L 的 function 裡面去
(I put this d sub i into function L here)

are code-mixed utterances in which Mandarin Chinese is the host language (L1) and English the guest language (L2).

By definition, code mixing is limited to bilingual speakers. It is estimated that there are more bilingual speakers in the world than monolinguals, and that the worldwide percentage of bilinguals is increasing [3]. Indeed, even in the United States, census data shows that 20% of those surveyed reported speaking a language other than English at home in 2007, compared to only 8% in 1980 [4]. Code-mixing is widely used among bilingual individuals, who when speaking their native language readily add words or phrases from a second language, especially English.

The technology for monolingual automatic speech recognition (ASR) is relatively mature, but that for code-mixed ASR is still in its infancy. This is reflected in downstream applications of automatic transcriptions. For instance, in lecture ASR, that is, ASR conducted on classroom lectures, even when lectures are often delivered in languages other than English, they often include occasional English words, and these English words are often keywords, which represent the most important content in an utterance. It is exactly this content then which is most relevant in terms of many spoken language understanding applications. However because of the heavily imbalanced nature of code-mixed lectures, ASR performance for these crucial English words is generally poor. Any improvements in code-mixed ASR, especially for English segments, therefore, will also benefit such downstream applications, and result in a more compelling online learning environment. This is also true for smartphone-based ASR applications: to appeal to a wider segment of the smartphone market, localization of such applications to areas other than the main English-speaking countries must take into account code-mixing.

One other problem with code-mixed ASR is the lack of training data. Monolingual data is expensive to acquire and transcribe; bilingual data is even more so. Resource-limited methods are needed that yield performance improvements even with relatively small amounts of training data.

In this paper, we propose an approach for the automatic recognition of code-mixed speech in which English-language segments are recovered within a multi-pass ASR framework (Figure 1). This is a speaker-dependent, resource-limited mechanism for the recovery of code-mixed speech which is related to what could be termed code-mixed language identification (CMLID). We first use a bilingual acoustic model, a bilingual lexicon, and a bilingual language model to generate the initial set of system recognition hypotheses (this baseline system is detailed in Section VI-B), after which we analyze these hypotheses and their audio signals (waveforms) to recover

A. Heidel is with the Department of Computer Science & Information Engineering, National Taiwan University, Taipei, Taiwan 10617 (R.O.C.) (e-mail: aaron@speech.ee.ntu.edu.tw).

H.-H. Lu (e-mail: r03942039@ntu.edu.tw) and L.-S. Lee (email: lslee@gate.sinica.edu.tw) are with the Department of Electrical Engineering, National Taiwan University.

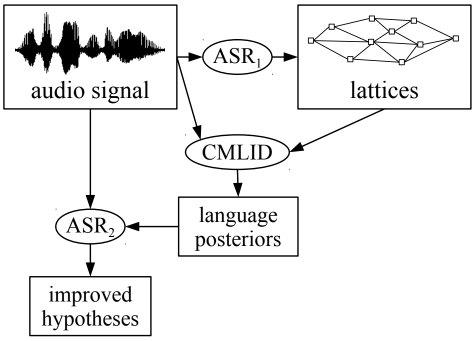


Fig. 1: Proposed system. Components ASR$_1$ and ASR$_2$ both use the same bilingual acoustic and language models and lexicon. The results of the first pass (ASR$_1$) are used to perform a targeted second pass (ASR$_2$).

misrecognized guest-language segments. We conduct our research on Mandarin-English code-mixed lecture recordings where Mandarin is the host language and English the guest language. Our task is to recover English-language segments that were misrecognized in the initial recognition pass.

We extend the work of Yeh et al., who use low-level acoustic-phonetic cues to enhance both Chinese- and English-language recognition, in particular focusing on bilingual acoustic modeling strategies involving state mapping [5] and frame-level CMLID using blurred posteriorgrams [6] to mitigate the effects of data imbalance. Here, we extend this work to higher levels and establish a comprehensive framework for English recovery. We focus on English segments, and use higher-level cues – phonotactic, prosodic, and linguistic in addition to acoustic-phonetic – to discriminate at the frame level between Chinese and English segments. We show that despite our focus on improving English recognition accuracy for this resource-limited task, our approach has the side-effect of additionally improving Chinese accuracy. We demonstrate considerable performance improvements not only with the traditional GMM-HMM recognition paradigm but also with a state-of-the-art hybrid CD-HMM-DNN recognition framework. Note that this approach is not limited to Chinese-English code-mixed speech but can be applied to any host-guest language pair.

II. Background and Related Work

A. Linguistic Background

For humans, the recognition of guest-language segments is a highly complex process [7]. In particular, bilingual subjects are known to take longer to recognize code-mixed words than words in the host language [8]; this could be due to the need to switch from the host to the guest model, or to wait for a longer context (presumably in search of higher-level cues to reduce the search space), or both.

Grosjean lists several effects that have been observed in human language processing [7]. He maintains that linguistic researchers must account for these effects when they theorize the structure and functions of “mixed language word recognition” models in the human brain. Among these effects are the following:

- *Frequency:* rare words take longer to recognize than common words [9]. This necessitates the use of linguistic cues.
- *Phonotactics:* “Words marked phonotactically as belonging to the guest language only ... are recognized sooner and with more ease than words not marked in this way.” [7]. Although phonotactic cues can be very effective, not every word can be distinguished in this way. This effect too must be resolved with higher-level cues.
- *Homophones:* Words in the guest language that are pronounced identically to words in the host language are more difficult to process than other guest-language words [7]. This shows that acoustic and phonotactic cues can mislead, and that higher-level (i.e., syntactic or linguistic) cues are needed sometimes to “break ties”.
- *Order independency:* In continuous speech, words are not always recognized one at a time; rather, words may be recognized simultaneously, or even in reverse order [10], [11]. A model that takes this into account cannot merely proceed from left-to-right, one word at a time, but must consider wider contexts, for instance, multi-word phrases or whole utterances, or even groups of utterances.

Hence we see that humans simultaneously use both high-level and low-level cues to recognize code-mixed speech. In this work we attempt to develop a similar albeit greatly simplified system for code-mixed ASR, drawing inspiration from human speech recognition in our search for useful cues and mechanisms.

B. Related Work

There is a wide body of research on language identification (LID) [12]–[14]. For general LID, the task has been to identify the language of a given spoken utterance. Recent NIST language recognition evaluations have specified utterance lengths of 30, 10, and 3 seconds, and have provided utterances produced by native speakers in their own languages. There are typically over 10 languages to choose from in deciding the language.

There are several important differences between CMLID and LID that preclude a straight transfer of LID methods to CMLID:

- *Language boundaries unknown:* In LID, the language boundaries are given, because the end of the sentence marks the end of a given language. In CMLID, however, the boundaries can fall anywhere within an utterance. Additionally, not every utterance is guaranteed to contain guest-language segments.
- *Short language segments:* In CMLID, speech segments for a language can be very short, sometimes less than a second long, whereas for LID, speech segments are at least three seconds long, depending on the task (usually the focus is on 30-second segments).
- *Non-native accents:* In CMLID, non-native accents are common in guest-language segments. Indeed, guest-language words are often pronounced in the style of the host language, using host-language phonemes and prosody. This can introduce severe pronunciation variations, which can significantly complicate the acoustic modeling task,

and grammar faults, which can make language modeling more difficult.

- *Binary choice of languages:* CMLID is simpler than LID only in its being limited to two possible languages as opposed to LID's ten or more.

Despite these differences, some LID techniques have been found to work well within a CM framework. In particular are phonotactic methods which leverage each language's constraints on phonotactic sequences such as CVC, which refers to a syllable like "bat" or "till" composed of a consonant-vowel-consonant sequence, or CVCC ("bats" or "tilt", consonant-vowel-consonant-consonant). Such syllables are perfectly natural in English but not generally in Mandarin, in which syllables do not end in non-nasal consonants. Also, some phonemes are language-specific: the ɪ vowel in *bit* occurs in English but not in Mandarin, and the umlaut ü vowel in 女, pronounced *nü*, occurs in Mandarin but not in English. Finally, the tonal quality of Mandarin can be leveraged with respect to English by the use of prosodic cues.

For Cantonese-English code mixing, Lee et al. perform language boundary detection (LBD). In addition to using biphone probabilities, they also use syllable-based and lattice-based methods, and demonstrate improved ASR results when LBD is applied to ASR [15], [16]. They further show that ASR confidences are unreliable around guest-language segments and should hence be augmented with LID information [17].

C. Frame GLD

We extend the work of Yeh et al. [6], who perform recognition of code-mixed lecture data using bilingual acoustic models and a bilingual language model and lexicon. They perform frame-level English detection (frame-level guest language detection, or frame GLD) within the conventional ASR framework, the output of which is used to boost the scores of detected English phoneme models during a second ASR pass. Following [18], they implement the detector as a neural network, but instead of using long-context MFCCs as input, they take their input from the first-pass recognition phoneme lattices, which are represented for each frame o_t as an N-dimensional posteriorgram vector $P_t = P(p_i|o_t), i = 1, 2, \ldots, N$, where p_i is a Chinese or English phoneme, N is the total number of phonemes for the two languages, and $P(p_i|o_t) = 0)$ for phonemes p_i that do not appear in the lattice at time t. From these phoneme posteriorgrams they extract "blurred" posteriorgram features (BPFs) as

$$P'(p_i|o_t) = \frac{P(p_i|o_t)^\beta}{\sum_k P(p_k|o_t)^\beta}, 0 < \beta < 1, \qquad (1)$$

where blurring factor β approaches 0. This blurring of the phoneme posteriorgrams yields increased sensitivity to low but nonzero phoneme posteriors, such as is common for the undertrained English phonemes, at $\beta = 0$ yielding a uniform distribution for all nonzero posteriors; the blur exponent thus dampens the effect of data imbalance. These BPFs are then used as input for the GLD neural network with the targets *English* and *non-English* (this includes both Chinese and silence phonemes).

Frame GLD outputs posterior probabilities $P(E|o_t)$ and $P(C|o_t)$ for English and Chinese (and silence) given each feature vector o_t for a speech frame at time t, where $P(E|o_t) + P(C|o_t) = 1$. Then the acoustic model score $P(q_j|o_t)$ for HMM state (i.e., senone) q_j given frame o_t is boosted to

$$\hat{P}(q_j|o_t) = \begin{cases} P(q_j|o_t) \times \frac{P(E|o_t)}{1-P(E|o_t)} & \text{if } P(E|o_t) > 0.5 \\ & \text{and } q_j \in \mathcal{E} \\ P(q_j|o_t) & \text{otherwise,} \end{cases} \qquad (2)$$

where $\hat{P}(q_j|o_t)$ is the score used in the recognizer and $\mathcal{E}$ is the set of all HMM states for English phoneme models. Thus if HMM state q_j is identified as an English senone, and if $P(E|o_t) > 0.5$, its score is increased according the detector's posterior probability $P(E|o_t)$; otherwise ($P(E|o_t) \leq 0.5$ or $q_j \notin \mathcal{E}$) the score remains unchanged. That is, if $P(C|o_t) > 0.5$, no action is taken, because the Chinese phoneme models are judged to have been trained on an amount of data sufficient to ensure acceptable performance.

In this work, we replace the frame GLD with a series of cascaded models that leverage both high- and low-level cues to yield improved posterior probabilities $P(E|o_t)$ and $P(C|o_t)$ for use in a targeted second ASR pass.

III. Framework

Our two-pass framework is shown in Figure 1: simply stated, we use the results of the first ASR pass to perform a targeted second ASR pass. That is, the audio signals are our training corpus on which we perform ASR using baseline acoustic models (AMs) and language model (LM) to generate the first-pass recognition results in the form of lattices. We then perform English-biased CMLID on the audio signals and the lattices to generate language posteriors, which we use in combination with the original audio signals to perform our second run of ASR (using the original AMs and LM), thus yielding an improved set of recognition hypotheses.

Figure 2 illustrates the different layers present in code-mixed speech. In this utterance, there are three Chinese segments (所以我省了很多, 或者, and 的空間) and two English segments (*bandwidth* and *bitrate*). Each layer is divided into different-sized tokens: white for silence (SIL), blue for Chinese (CH), and striped red for English (EN). The higher the layer, the longer the tokens. The word layer at the top, which contains the longest tokens, is composed of English words and Chinese multi-character pseudo-words. Note that in Chinese, word boundaries are ill-defined: although characters often do have definite collocations, these collocations are fluid and can change significantly for different topics or modes of discourse. Below the word layer, the syllable layer is composed of single characters and English syllables, all of which are sequences of consonants and vowels. Below the waveform is the phone layer, which is composed of the various consonants and vowels that make up each syllable; in this work these correspond to hidden Markov model (HMM) monophones. The phone labels shown (sil, s, u, o, #, @, AE, IH, TH, etc.) are those used in the ASR system, and reflect the implicitly bilingual composition of the ASR models: they model a full set of Chinese phones as well as a full set of English phones. Between the phone

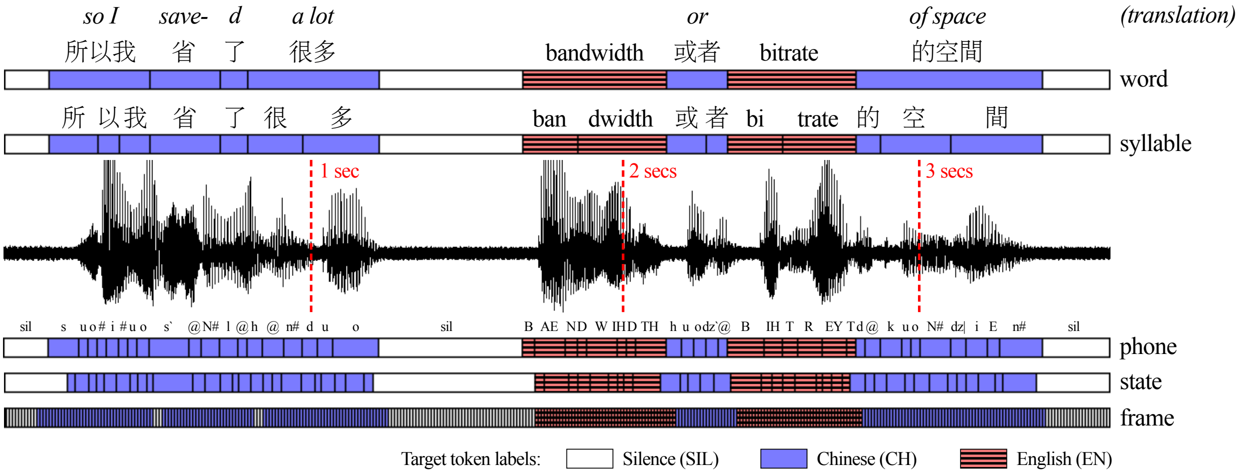


Fig. 2: The audio signal and the main layers present in code-mixed speech (translation: "*So I saved a lot of space in* bandwidth *or* bitrate"). For each layer, the task is to classify each token in the sequence given the features associated with that token. Identity features are shown for word, syllable, and phone layers.

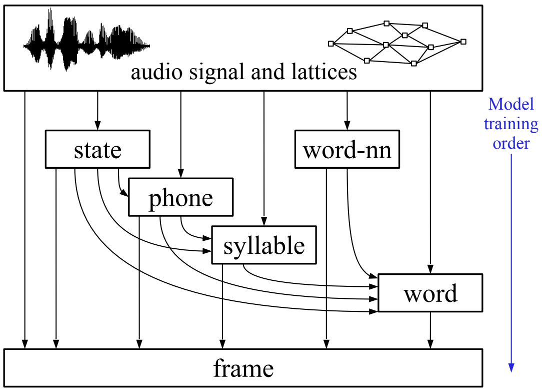


Fig. 3: The six CMLID layers. Arrows show input and output for each component. All layers are linear-chain CRF models except word-nn which is a recurrent neural network.

and frame layers is the state layer, which is not an HMM state but is generated using a different method, as described in Section IV-B. At the bottom is the frame layer, composed of 10ms tokens. Note that the frame-layer tokens are all equal in length, but the token lengths for other layers depend on the contents of the audio signal.

The CMLID module from Figure 1 is composed of the various cascaded layers shown in Figure 3, each of which is used to classify a different level of speech data. The incoming and outgoing arrows in this diagram show the input and output for each layer. For instance, the syllable layer takes input from the preceding state and phone layers, as well as the audio signals and lattices, and its output is used as input for the word and frame layers.

For each layer, the task is to decide what class (SIL, CH, or EN) each token belongs to, given the entire token sequence in the utterance as well as the features for each token. These classes are shown in Figure 2 as the different-colored target tokens. We include silence as a class because silences often precede or follow code switches, and thus are viewed as communication that can yield usable cues for our purposes. Thus for the word layer, we seek to determine the class for each word in an utterance, given the features of each word; likewise for the syllable, phone, state, and frame layers.

The input for the frame layer is derived from the audio signals and lattices, and also includes cascaded features from all of the higher layers: state, phone, syllable, word-nn, and word. The output of the frame layer is then used as input to boost English phonemes during the second ASR pass.

A. *Models*

1) Conditional random field: We model the state, phone, syllable, word, and frame layers using linear-chain conditional random fields (CRFs) [19], [20]. Because they are discriminative, sequential models, linear-chain CRFs are well suited to the task at hand. A linear-chain CRF is a distribution $p(\mathbf{y}|\mathbf{x})$, where $\mathbf{y}$ and $\mathbf{x}$ are state and observation sequences, respectively. (Note the 'state' mentioned here is a CRF state and is unrelated to the CMLID state layer.)

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\}, \qquad (3)$$

where

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\} \qquad (4)$$

is a normalization factor, $\Lambda = \{\lambda_k\} \in \Re^K$ is the parameter vector to be learned, and $\{f_k(y, y', \mathbf{x}_t)\}_{k=1}^{K}$ is a set of real-valued functions.

As shown in Figure 2, the possible output states y for our CRF models are SIL, CH, and EN. Thus, given $P_{fr}(y_t|x_t)$, the output of the final CRF frame model for frame t is

$$P'(E|o_t) = P_{fr}(\text{EN}|x_t) \quad \text{and} \qquad (5)$$

$$P'(C|o_t) = 1 - P_{fr}(\text{EN}|x_t). \qquad (6)$$

As with frame GLD in Section II-C, we use this $P'(E|o_t)$ and $P'(C|o_t)$ to boost corresponding English phonemes during the second ASR pass.

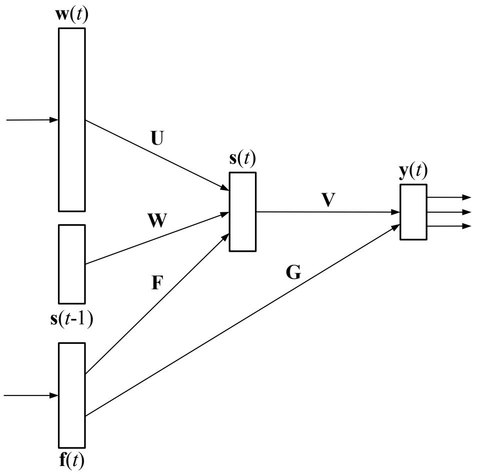


Fig. 4: Modified context-dependent RNNLM diagram. The original RNNLM class+word output layer is simplified to SIL, CH, and EN.

2) Recurrent neural network: For the word-nn layer, we use a modified context-dependent recurrent neural network language model (RNNLM) [21], [22] in which the factorized (class+word) output layer [23] is simplified to just the three output classes SIL, CH, and EN.

This is shown in Figure 4, in which $\mathbf{w}(t)$ is the 1-of-N word vector for time t, $\mathbf{f}(t)$ represents the various continuous-valued auxiliary feature vector corresponding to $\mathbf{w}(t)$, $\mathbf{s}(t-1)$ is the previous word's RNN state vector, $\mathbf{s}(t)$ is the current word's state vector, and $\mathbf{y}(t)$ is the output vector. In our case $\mathbf{y}(t) \in \{\text{SIL}, \text{CH}, \text{EN}\}$. $\mathbf{U}$, $\mathbf{W}$, $\mathbf{F}$, $\mathbf{V}$, and $\mathbf{G}$ are the various synapse matrices. In particular, $\mathbf{G}$ represents a simple maximum entropy model with N-gram features [24].

The state and output layers are computed as

$$\mathbf{s}(t) = f\left(\mathbf{U}\mathbf{w}(t) + \mathbf{W}\mathbf{s}(t-1) + \mathbf{F}\mathbf{f}(t)\right), \tag{7}$$

$$\mathbf{y}(t) = g\left(\mathbf{V}\mathbf{s}(t) + \mathbf{G}\mathbf{f}(t)\right), \tag{8}$$

where $f(z)$ is the sigmoid activation function and $g(z_m)$ is the softmax function:

$$f(z) = \frac{1}{1+e^{-z}} \quad \text{and} \quad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}. \tag{9}$$

The model is trained by using stochastic gradient descent to find the weight matrices $\mathbf{U}$, $\mathbf{W}$, $\mathbf{F}$, $\mathbf{V}$, and $\mathbf{G}$ such that the likelihood of the training data is maximized.

IV. Features

The strength of the proposed approach lies in the combination of both low- and high-level features, and the selection of the most effective of these features, which yields a small set of nonlinear, high-order features highly targeted to English-specific CMLID.

A. CRF Topology

1) Feature binning: Because we use discrete features in our CRF models, we perform uniform binning on numeric features and exponential binning on duration-type features. Thus a real-valued feature like `conf`, $0 \leq$ `conf` ≤ 1, is set to one of the discrete values `bin0`, `bin1`, or `bin2` corresponding to

$$\left[0, \frac{1}{3}\right], \left(\frac{1}{3}, \frac{2}{3}\right], \left(\frac{2}{3}, 1\right].$$

For a durational feature like `len` (length in frames), $0 \leq$ `len` ≤ 800, we have `bin0`, ..., `bin10` corresponding to

$$\left[\frac{800}{2^{11}}, \frac{800}{2^{10}}\right], \left(\frac{800}{2^{10}}, \frac{800}{2^{9}}\right], \ldots, \left(\frac{800}{2^{1}}, \frac{800}{2^{0}}\right],$$

which is equivalent to uniform binning for log features.

label	CH	SIL	EN	EN	CH	CH	EN	EN	CH
t	8	9	10	11	12	13	14	15	16
syl	多		ban	dwidth	或	者	bi	trate	的
cv	CVV	0	CVC	CCVCC	CVV	CV	CV	CCVC	CV
len	25	47	18	29	13	7	18	24	8
conf	1.00	1.00	.72	.60	.77	.64	.93	.95	1.00

Fig. 5: Syllable-level CRF target labels and sample feature vectors for part of the utterance in Figure 2. Feature `syl` is the syllable's lexical identity, `cv` is its consonant-vowel sequence, `len` is its length in 10ms frames, and `conf` is its lattice-based confidence.

2) Atomic features: Figure 5 illustrates the observations and labels of a CRF model. For each token, we have listed the features `syl`, `cv`, `len`, and `conf`. Features that we use in practice include not only *atomic features* but also *feature conjunctions* [25]. With respect to the syllable token at $t = 11$, the atomic feature `cv[0]` is "CCVCC". Thus features like `cv[-1]`="CVC" or `cv[1]`="CVV" correspond to the same token but contain contextual information, specifically regarding the previous (*ban*) or next (或) tokens, respectively. In this way we denote contextual features for any given token.

Where $\mathcal{F}_a$ is the set of discrete values that feature a can take, feature rule $a[i]$ during CRF training is expanded to $|\mathbf{y}| \times |\mathbf{y}| \times |\mathcal{F}_a|$ CRF feature functions $f_k(y, y', \mathbf{x}_t)$ (see Eq. 3). Thus feature rule `len[0]` would expand to $3 \times 3 \times 11 = 99$ feature functions.

3) Feature conjunctions: Feature conjunctions are discrete concatenations of two or more features. For instance, $\langle$`cv[0]`+`cv[1]`$\rangle$ is the conjunction of the `cv` feature for the current and next tokens: for $t = 11$, $\langle$`cv[0]`+`cv[1]`$\rangle$ = "CCVCC.CVV"; note that in CRF training, this is seen as a single discrete value. Heterogeneous conjunctions are also used: $\langle$`syl[0]`+`cv[-1]`$\rangle$ = "dwidth.CVC". (Note that due to binning, for $\langle$`len[0]`+`cv[0]`$\rangle$ we would have something like "len.11_04.CCVCC".) A conjunction $\langle a[i]+b[j]\rangle$ of features a and b expands to $|\mathbf{y}| \times |\mathbf{y}| \times |\mathcal{F}_a| \times |\mathcal{F}_b|$ CRF feature functions.

4) Feature functions, rules, and groups: To reduce complexity, during feature selection, we select features at the feature group level and not at the feature function level. A feature group for feature `a` includes the following 14 feature rules: `a[-2]`, `a[-1]`, `a[0]`, `a[1]`, `a[2]`, $\langle a[-3]+a[-2]\rangle$, $\langle a[-2]+a[-1]\rangle$, $\langle a[-1]+a[0]\rangle$, $\langle a[0]+a[1]\rangle$, $\langle a[1]+a[2]\rangle$, $\langle a[2]+a[3]\rangle$, $\langle a[-2]+a[0]\rangle$, $\langle a[-1]+a[1]\rangle$, and $\langle a[0]+a[2]\rangle$.

A feature group for the conjunction of features a and b includes the following 23 feature rules: $\langle a[-2]+b[-2]\rangle$,

$\langle a[-1]{+}b[-1]\rangle$, $\langle a[0]{+}b[0]\rangle$, $\langle a[1]{+}b[1]\rangle$, $\langle a[2]{+}b[2]\rangle$, $\langle a[-3]{+}b[-2]\rangle$, $\langle b[-3]{+}a[-2]\rangle$, $\langle b[-2]{+}a[-1]\rangle$, $\langle a[-2]{+}b[-1]\rangle$, $\langle b[-1]{+}a[0]\rangle$, $\langle a[-1]{+}b[0]\rangle$, $\langle b[0]{+}a[1]\rangle$, $\langle a[0]{+}b[1]\rangle$, $\langle a[1]{+}b[2]\rangle$, $\langle b[1]{+}a[2]\rangle$, $\langle b[2]{+}a[3]\rangle$, $\langle a[2]{+}b[3]\rangle$, $\langle b[-2]{+}a[0]\rangle$, $\langle a[-2]{+}b[0]\rangle$, $\langle a[-1]{+}b[1]\rangle$, $\langle b[-1]{+}a[1]\rangle$, $\langle a[0]{+}b[2]\rangle$, and $\langle b[0]{+}a[2]\rangle$.

Thus feature groups like `len` or ⟨`conf+len`⟩ are composed of feature rules like `len[-1]` or ⟨`conf[2]+len[3]`⟩, each of which expands during CRF training to feature functions of the form $f_k(y_t, y_{t-1}, \mathbf{x}_t)$, each of which corresponds to a CRF parameter λ_k.

The CRF toolkit we use supports both L1 and L2 regularization [26]. We use L1 regularization to minimize model size and reduce training times. Also, only those feature functions are considered that occur in the training corpus no less than three times.

B. Token Durations

As shown in Figure 2, token durations depend on the layer. From the 1-best hypotheses in the first-pass recognition lattices, we extract token durations for the word, syllable, and phone layers.

Token durations in the state layer are based not on recognition results but on a hierarchical agglomerative clustering of the waveform-derived MFCCs into acoustically stationary segments [27]. This yields tokens that are slightly shorter on average than those in the phone layer. When thus generating state tokens in a manner orthogonal to that of the phone layer, the hope is to counteract any acoustic model biases, and to complement features in the phone layer.

C. Feature Types

The success of this framework for English segment recovery depends on the strength of the features used. To that end, we generate many different features and try all possible feature conjunctions to see what works best. For each layer we extract a different set of features, but many feature types are shared across layers. The foundational unit for all layers is the blurred posteriorgram feature (BPF) from Section II-C, extracted as in [6] using a blur factor of 0.01.

1) Identity features: These lexical identities include phoneme labels `phoneme1` and `phoneme2` for the state and phone layers: `phoneme1` is the highest-scoring phoneme over the span of the state or phone token, and `phoneme2` is the second-highest scoring phoneme. In the syllable layer, `syl.orig` is a concatenation of the phonemes over the token span (for the syllable *see* this would be "S+IY"), and `word` in the word layer is the corresponding word in the underlying 1-best hypothesis. The word layer also contains `word.syllables`, which like `syl.orig` is a concatenation of the the phonemes over the word's token span ("B+AE+N,D+W+IH+D+TH" for *bandwidth*).

2) Duration features: Duration features like `len`, `len.syl`, and `len.word` record the length of the token in 10ms frames. For the syllable layer, `len.onset`, `len.nucleus`, and `len.coda` record the length of the three syllable components. As shown in Figure 2 (ban-dwidth vs. band-width, bi-trate vs. bit-rate), the syllabification of English words differs from standard dictionary syllabifications. This is because it was generated using the maximum onset principle, which maximally assigns consonants to the beginning (onset) of the following syllable before assigning them to the end (coda) of the previous syllable. We do not take into account compound words. In Chinese, syllabification is trivial due to the simpler structure of Chinese syllables.

In the word layer, `len.silence.before` and `len.silence.after` record the lengths of any preceding or following silence tokens. Also, `len.syl.avg` records the average syllable length for this word, and `avg_ph_in_syl` the average number of phones in each syllable in the word. Related to duration features are the word layer's `phones` and `syllables`, which record the number of phonemes and syllables that this word contains, respectively.

3) Confidence features: The confidence features `conf` and `entropy` are measures of lattice confidence. `conf` is set to the mean weight of `phoneme1` over the BPF's covering the token, and `entropy` is the average entropy of the BPF distributions covering the token.

4) Positional features: These features record where in the utterance the token is, and include `fpos` (distance from utterance head), `bpos` (distance from tail), and `mpos` (distance from middle).

5) Distributional features: These features measure the mean and standard deviations of various other features: `latl` and `ratl` measure the exponential moving average of the token lengths to the left and right of the current token, and `lstd` and `rstd` measure their exponential moving standard deviations. In the syllable layer, we additionally record distributional features for onset, nucleus, and coda lengths, and in the word layer, we include distributional measures for the surrounding tokens' `phones` and `syllables` features.

6) Articulatory features: We apply linguistic knowledge to extract additional language-independent information from the lattices in the form of articulation features `sono` and `cv`. These are respectively fine and coarse measures of phoneme sonorance. Feature `sono` is composed of the following classes: voiced obstruent, voiceless obstruent, voiced sonorant, voiceless sonorant, vowel, and silence. (Stops, affricates, and fricatives are considered obstruents, and nasals and approximants (or glides) are considered sonorants.) Feature `cv` is composed of consonant, vowel, and silence. These distinctions were motivated in part by the work of Yin et al. on voiced/unvoiced duration modeling [28], and also in hopes of reflecting language-specific phonotactic constraints. We also generate `place` to distinguish between silence, front, central, and back sounds, and `voicing` for silence, voiced, and voiceless sounds.

For syllables, in addition to the `syl.orig` identity feature, we describe each syllable's articulatory makeup such as with the `syl.cv` feature from Figure 5, which is a concatenation of Cs and Vs corresponding to each constituent phoneme's CV class. We likewise use `syl.manner`, `syl.place`, and `syl.sono` to describe syllable makeups in terms of phoneme manner and place of articulation, and in terms of phone sonorance (per `sono`).

Note that all articulation features are in fact BPF-based expectations over the corresponding token duration.

7) Prosodic features: For the syllable layer, we extract prosodic features that reflect various measures of syllable pitch and energy. We follow [29] and extract 20 pitch features (`pitch_*`) and 13 energy features (`energy_*`). These features are extracted directly from the corresponding audio signal. For the state and phone layers, we extract the same prosodic features but treat the underlying state and phone tokens as pseudo-syllables. For the word and word-nn layers, we extract prosodic features for the leftmost and rightmost syllables of the word, yielding `left.pitch_*`, `left.energy_*`, `right.pitch_*`, and `left.energy_*`.

8) Features for word-nn layer: The word-nn neural network model uses the groundtruth transcripts for training and lattice 1-bests for testing. Word boundaries are extracted from the results of forced alignment. Features used for each word include the word itself, its lattice-derived confidence `conf`, duration features `len.word`, `len.silence.before`, `len.silence.after`, and `len.syl.avg`, `syllables`, `phones`, `avg_ph_in_syl`, `fpos`, `mpos`, and `bpos`. This is in addition to the word-level prosodic features. Note that these features are not binned as with the CRF model but are used directly. Due to the order independency effect mentioned in Section II-A, we train both left-to-right and right-to-left models, and use as the output of this layer the average of the testing results of both models.

9) Linguistic features: The language model backoff behavior `lmbb` is a token-level variant of that proposed by Fayolle et al. [30]. Based on the utterance's probability chain given a token-based 20-gram language model, `lmbb` is the backoff level of the corresponding probability within the chain. For example, if an explicit parameter for the full 20-gram of a given token segment's probability is present in the LM, its `lmbb` is set to 20; if the highest-order explicit parameter for the probability is a 3-gram, then `lmbb` is set to 3. Thus `lmbb` reflects the language model's confidence with respect to the token segment in context.

For the state and phone layers, `lmbb` is generated based on the identity `phoneme1`. In the syllable layer, `lmbb.cv`, `lmbb.sono`, `lmbb.manner`, and `lmbb.place` are additionally generated based on their corresponding articulatory syllable representations.

For the word layer, word-nn class marginals are used as linguistic cues.

10) Cascaded features: These are the output class marginals from one layer that are used as input features for another layer. When the token boundaries between the layers are different, the expectation is taken over the target layer's token durations. They are generated in the form **LAYER**.`marg0`, **LAYER**.`marg1`, and **LAYER**.`marg2`, for the SIL, CH, and EN class marginals, respectively, where **LAYER** specifies the source layer.

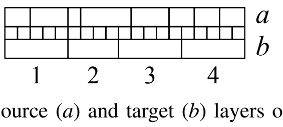


Fig. 6: Example of source (*a*) and target (*b*) layers of cascaded features.

Cascaded features also include **LAYER**.`tc`, **LAYER**.`atl`, and **LAYER**.`marg_summary`. Referencing Figure 6, **LAYER**.`tc` records the number of source (layer *a*) tokens that are part of the corresponding target (layer *b*) token, and **LAYER**.`atl` records the mean token length for those same source tokens. Thus **A**.`tc` for *b*[1–4] is $\{2, 2, 3, 3\}$, and **A**.`atl` is $\{2.5, 2.5, 3, 2\}$. **LAYER**.`marg_summary` is a textual sorted summary of the three numeric marginal features after coarse quantization.

11) CRF target labels: The CRF target labels are generated similar to the way cascaded features are generated, by treating the forced alignments of the ground-truth word transcriptions as the source level and recording the maximum resultant marginal class as the target label for the corresponding token.

Thus the longer the target tokens – most notably in the word layer – the greater the chance that errors will propagate.

D. Feature Counts

Using all possible feature rules (see Section IV-A4) results in large CRF models: 20M feature functions for the state layer (estimated without L1 regularization but with a count cutoff of 3), 22M for the phone layer, 120M for the syllable layer, 240M for the word layer, and more for the final frame layer (exceeded available memory).

E. CRF Feature Induction

Using all possible feature rules not only results in intractable CRF models, but it is also no guarantee of better models even if they were tractable. Hence the need for feature induction, where we select from a pool of potential features (that is, feature groups) an ensemble of complementary features: features that are good not only individually but also when used together.

McCallum's CRF feature induction scheme [25] operates on the assumption that previously-induced feature weights remain unchanged when new features are added. However, this assumption does not hold in practice: two features used together can yield performance worse than when either feature is used alone.

In this work we use a simple scheme for CRF feature group induction called greedy feature selection (GFS), in which we iteratively choose the yet-unchosen best-performing feature group for inclusion in the ensemble e of final feature groups. This is detailed in Algorithm 1.

Algorithm 1 GFS

```
ensemble ← ∅
pool ← all feature groups
repeat
    maxscore ← 0
    for each group in pool do
        score ← evaluate(group)
        if score > maxscore then
            maxscore ← score
            bestgroup ← group
        end if
    end for
    ensemble ← ensemble ∪ bestgroup
    pool ← pool − bestgroup
until done
```

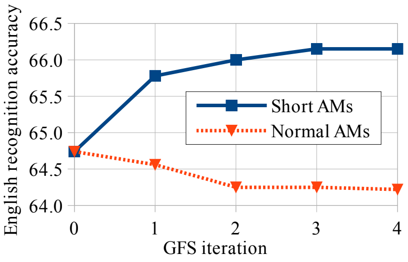


Fig. 7: Development set English recognition accuracies for short AMs and normal AMs in preliminary experiments. Shows the effect of k-way cross validation.

As the scheme is exact, in that every feature group is evaluated, it is more CPU-intensive than McCallum's method. It is, however, easily parallelizable over different machines and multiple cores, unlike McCallum's method, for which no parallel implementation is publicly available. The GFS objection function evaluate() is a weighted mean of the 1-versus-all soft class F-measures. In this work, we use a weight vector of (SIL = 0, CH = 0, EN = 1) to recover English segments, which are under-represented in the training data. That is, we select a feature group based solely on how well it classifies English tokens.

F. *Generating Cascaded Features*

One problem when using cascaded features for model training is how to reduce training/test set mismatch. The naive approach would be to simply train the model on the training set, and then test the model on the same training set, and use the results as features for the next layer. With this approach, however, the model has already seen the data, and does unreasonably well: much better than it would on unseen data, such as the development or test sets. Cascaded features generated in this way are unreasonably strong: during GFS they do well, but during testing they yield poor performance. Hence GFS does not choose the best features, being misled by the unrealistic performance of the cascaded features.

To ensure that cascaded features are not unreasonably strong, we use k-way cross validation to generate cascaded features. We prepare one corpus for each of the 15 days of lectures represented in the training corpus. That is, to generate the ASR first-pass lattices from which we extract features for model training, we divide the training set into $K = 15$ partitions $p_k, k = 1, 2, \ldots, K$, and define for each short AM θ_k a corpus composed of all partitions but p_k. Thus we train each short AM θ_k on everything but p_k in the training corpus. We then use θ_k to generate the recognition lattices for the utterances within p_k, yielding a set of training set word lattices that are better matched to the dev and test set lattices.

Note that this extra step is unnecessary for approaches that use no cascaded features, such as the frame GLD baseline described in Section II-C.

We also use this approach with CRF models to extract cascaded features for later CRF models. As shown in Figure 7, preliminary experiments showed considerable performance improvements using this approach, so all experiments reported here were conducted using such a cross validation-like approach for cascaded features.

V. Layers

As shown in Figure 3, we start by running GFS on the state layer, considering identity, duration, confidence, positional, distributional, articulatory, pseudo-prosodic, and linguistic features. We train the state model using the resultant ensemble of feature groups. Next we run GFS on the phone layer, considering the same types of features as in the state layer but with the addition of the cascaded state layer features. We continue on likewise with the syllable layer, which takes into account now fully prosodic features in addition to the same types of other features, as well as the cascaded state and phone layer features. The word-nn layer is trained on the features listed in Section IV-C8, independently of all other layers. For the word layer, we consider prosodic cues from the left- and rightmost syllables of each word in addition to the same types of other features, as well as the cascaded state, phone, syllable, and word-nn features.

Finally, we perform late fusion with the lowest-level frame model, as we run GFS on a pool of cascaded features from all of the other layers in addition to native frame-level identity, confidence, distributional, and positional features. The resultant feature group ensemble yields the final model with which we generate language posteriors for use in running a targeted second recognition pass.

A. *DNN frame model*

For comparison with the final CRF model, that is, the late fusion frame-based model, we conduct additional experiments using deep neural network (DNN) models [31]. We use standard DNN models, with the sigmoid function for internal activations and softmax for the last layer. All models have a dropout rate of 50% [32]. We found that although using dropout requires nearly twice the time as models without dropout, dropout yields significant performance improvements.

This DNN frame model is trained on the same input as the CRF model. However, because of the numeric nature of the DNN model, all numeric CRF features are expressed as real numbers instead of bins, and all discrete features are expressed as 1-of-N binary feature vectors. For the final frame-level model, this conversion from CRF to DNN results in 572 numeric features per frame.

Recall that the CRF model takes into account features from up to 3 frames previous and 3 frames following (Section IV-A4). Thus we use a matching input context for the DNN models: an input supervector including the features for the previous 3 frames, the current frame, and the following 3 frames. The tuned DNN topology for this layer is thus 572 × 7 = 4004 nodes for the input layer; 8000, 4000, 2000, 1000, and 500 nodes for the five internal DNN layers; and 3 nodes (0 for SIL, 1 for CH, and 2 for EN) for the output layer.

The DNN frame LID model for this paper was trained using libdnn [33].

TABLE I: *DSP training corpus statistics. Under 'utterances', 'English' refers to utterances containing both Chinese and English words. Language segments are contiguous sequences of words in a given language.*

	Utterances	Language segments	Words
Total	9174	20448	79652
Chinese	4538 (49%)	13647 (67%)	70162 (88%)
English	4636 (51%)	6801 (33%)	9490 (12%)

VI. Experimental Setup

A. NTU Lecture Corpus

We perform code-mixed speech recognition on Mandarin classroom lectures delivered at National Taiwan University. Table I shows the extent of data imbalance for words, language segments, and utterances. The average English segment length is 1.3 words, or 0.75 seconds. On average, each utterance contains 8.7 words and is 3.5 seconds in length. The training set is 9 hours long, and the dev and test sets both contain approximately 2200 utterances and are 2.5 hours long in total.

B. Baseline recognition systems

We evaluated the proposed approach on two systems: a GMM-HMM system and a hybrid CD-HMM-DNN system.

We calculated recognition accuracy as a combination of English word-based accuracy and Chinese character-based accuracy [34].

1) GMM-HMM system: The GMM-HMM system was a standard HMM/Viterbi-based system. We used speaker-dependent, GMM, 3-state, triphone acoustic models (AMs), a modified Kneser-Ney trigram language model (LM), and a 13K-word lexicon composed of English words, Chinese multi-character words extracted using PAT trees [35] from a large corpus, and Chinese characters. The LM was an interpolation of a well-trained broadcast news background model trained on Mandarin Gigaword and a model trained on the lecture corpus training set. Note that this baseline system is implicitly bilingual, in that it was trained on bilingual data: the acoustic models contain a complete set of Chinese phones and a complete set of English phones, the lexicon contains both Chinese and English words, and the language model was trained on code-mixed data.

2) Hybrid system: For the hybrid (CD-HMM-DNN) system, the DNN AM was trained on MFCC features with 4 frames of context (9 frames total) for an input dimension of 351, and outputted 6073 senone probabilities. The senone labels for DNN training were determined using forced alignment results from the GMM-HMM system. The DNN contained 4 hidden layers, each with 2048 nodes, and used sigmoid activation functions. The model was initialized using random weights. During training, the learning rate started at 0.005, and once the development set improvement fell below 1% absolute, the learning rate was reduced by a factor of 0.9 after each epoch. The training used a batch size of 256 and was concluded (early stop) once the development set improvement fell below 0.05% absolute.

After training was completed, decoding was accomplished by feeding the DNN AM-emitted senone probabilities to the Viterbi decoder. This decoder used the same LM and lexicon used in the GMM-HMM system.

The acoustic models for the hybrid system were trained using the Kaldi toolkit [36].

C. Baselines and upper bounds

Our experiment baselines were each system's first-pass recognition results. In Table II we list the respective baseline development set recognition accuracies. Additionally, for the GMM-HMM system, we sought to improve on the frame GLD results [6].

TABLE II: *Baseline development set recognition accuracies. 'CH', 'EN', and 'OV' are Chinese, English, and overall accuracies.*

System	CH	EN	OV
GMM-HMM	82.9	62.4	81.3
Hybrid	87.5	79.4	86.8

For the GMM-HMM system, for further comparison, we trained a baseline MFCC-DNN LID classifier which takes as input the frame-level MFCCs to classify each frame as either silence, Chinese, or English. This long-context baseline took as input 81 frames of MFCC features: 40 previous frames, 1 current, and 40 following, making for 0.81 seconds of context. The baseline MFCC-DNN LID model thus had a 3159-node (81 frames × 39 MFCCs/frame) input layer, and three internal layers with sizes of 4000, 1024, and 1024. During training we used dropout rates of 50%. This baseline represents the conventional approach to LID, to be used before performing full recognition of code-mixed utterances, and as such was different from the frame GLD approach, which as mentioned in Section II-C is a multi-pass technique which takes its input from first-pass recognition phoneme lattices.

For both GMM-HMM and hybrid systems, oracle experiments show the upper performance bounds for any schemes that use this language posterior-based frame-boosting framework for rescoring. The oracle uses the ground-truth language posteriors, as defined by the results of forced-alignment using acoustic models from the GMM-HMM system.

VII. Experimental Results

We conducted experiments on both the GMM-HMM and hybrid systems.

A. GMM-HMM system

The results of GFS for the GMM-HMM system are shown in Tables III and IV. Table III shows the development set CRF model F-measures for each class for each GFS iteration for each layer: in the '# fea' column is listed the number of discrete CRF feature functions, and in the 'Feature groups' column is listed the feature group chosen for addition to the feature group ensemble during that GFS iteration. Table IV shows the recognition accuracies for the development set when the language posteriors from the CMLID module are used for a targeted second recognition pass: for each GFS iteration, the Chinese, English, and overall accuracies are shown, as well as their corresponding deltas with respect to the baseline

TABLE III: *GMM-HMM system: GFS feature group ensembles, feature function counts, and development set CRF F-measures. Features such as* **PH**.`marg2` *or* **STATE**.`marg_summary` *are cascaded features from earlier layers. '# fea' is the number of CRF feature functions (Eq. 3) for the iteration.*

Layer	GFS iteration	Feature groups (cumulative)	# fea	SIL	CH	EN
State	1	⟨`phoneme1`+`phoneme2`⟩	66208	0.473	0.874	0.723
	2	+ `phoneme1`	77734	0.477	0.875	0.737
	3	+ ⟨`conf`+`lat1`⟩	80542	0.483	0.877	0.746
Phone	1	⟨`phoneme1`+`phoneme2`⟩	51510	0.444	0.889	0.718
	2	+ `phoneme1`	64333	0.449	0.892	0.734
	3	+ ⟨`conf`+`bpos`⟩	65865	0.448	0.893	0.742
Syllable	1	⟨`syl.orig`+**PH**.`marg2`⟩	23565	0.626	0.887	0.703
	2	+ ⟨**STATE**.`marg2`+**PH**.`marg2`⟩	24332	0.628	0.890	0.723
	3	+ ⟨`syl.orig`+**STATE**.`marg_summary`⟩	44790	0.631	0.892	0.734
Word	1	⟨**STATE**.`marg_summary`+**PH**.`marg1`⟩	10417	0.856	0.872	0.573
	2	+ ⟨`wd.syllables`+**PH**.`marg_summary`⟩	27578	0.870	0.880	0.595
	3	+ ⟨`left.energy_max`+**SYL**.`marg2`⟩	27567	0.872	0.883	0.602
Frame	1	⟨`phoneme1`+**SYL**.`marg_summary`⟩	147603	0.625	0.843	0.691
	2	+ ⟨`phoneme2`+**STATE**.`marg2`⟩	177573	0.623	0.843	0.700
	3	+ `phoneme1`	202638	0.629	0.845	0.706
	4	+ ⟨**PH**.`awl1`+**PH**.`marg_summary`⟩	201599	0.633	0.846	0.709
	5	+ ⟨`conf.lat1`+**WD_NN**.`marg0`⟩	205666	0.635	0.845	0.712
	6	+ ⟨**SYL**.`marg0`+**SYL**.`marg1`⟩	214297	0.636	0.846	0.716
	7	+ ⟨`fpos`+**STATE**.`marg1`⟩	211958	0.638	0.845	0.714

TABLE IV: *GMM-HMM system: Development set recognition accuracies. Statistical significances and deltas expressed as percentages. 'CH', 'EN', and 'OV' are Chinese, English, and overall accuracies.*

	GFS	ASR accuracy				CH	EN	OV
Layer	iter.	CH	EN	OV	Sig	Δ	Δ	Δ
Baseline		82.9	62.4	81.3				
State	1	83.3	67.3	82.0	98.0	0.5	7.9	0.9
	2	83.3	67.5	82.1	98.6	0.5	8.2	1.0
	3	83.4	67.7	82.2	98.9	0.6	8.6	1.0
Phone	1	83.4	67.4	82.1	97.9	0.6	8.1	1.0
	2	83.4	67.0	82.1	97.3	0.6	7.4	1.0
	3	83.4	67.6	82.2	98.7	0.6	8.3	1.1
Syllable	1	83.3	67.0	82.0	97.9	0.5	7.3	0.9
	2	83.4	66.6	82.1	96.7	0.6	6.8	0.9
	3	83.4	67.0	82.1	97.4	0.6	7.3	1.0
Word	1	83.4	66.7	82.1	96.7	0.6	6.9	1.0
	2	83.6	66.4	82.2	96.5	0.8	6.5	1.1
	3	83.5	66.6	82.2	97.9	0.7	6.8	1.1
Frame	1	83.4	67.2	82.1	98.2	0.6	7.7	1.0
	2	83.4	68.0	82.2	99.2	0.6	9.0	1.1
	3	83.4	68.1	82.3	99.1	0.7	9.2	1.2
	4	**83.4**	**68.2**	**82.3**	**99.3**	**0.7**	**9.3**	**1.2**
	5	83.5	68.0	82.3	99.2	0.7	9.0	1.2
	6	83.4	68.1	82.3	99.5	0.7	9.2	1.2
	7	83.5	68.1	82.3	99.5	0.7	9.2	1.2

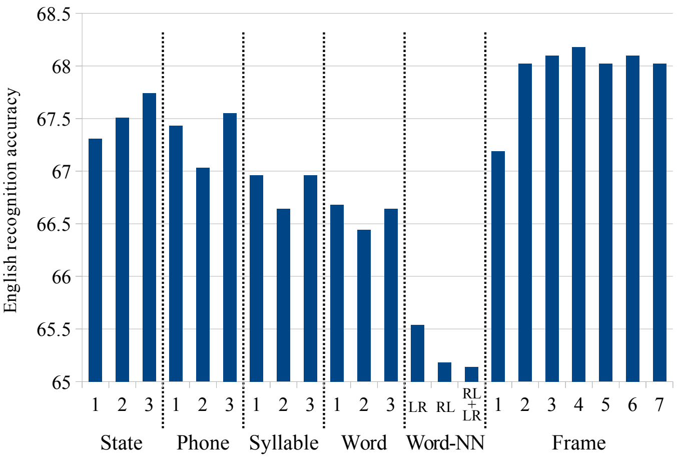


Fig. 8: GMM-HMM system: Development set English recognition accuracies.

accuracies, along with the statistical significance for each iteration. The significances shown are for English recognition accuracy over the baseline.

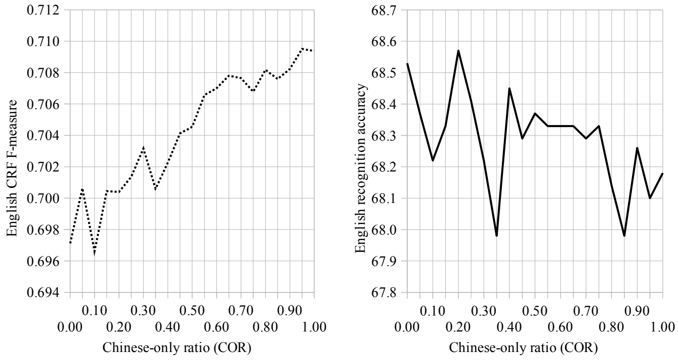
As for the feature groups chosen by GFS, we note that lexical identity cues such as `phoneme1` and `phoneme2` contain important information for classification for both state and phone layers, as do `syl.orig` and `wd.syllables` for the syllable and word layers. Also, cascaded features are shown to be useful as well. In the word layer, the inclusion of `left.energy_max` feature shows that word prosody discriminates between Mandarin and English, specifically, the energy of a word's beginning syllable.

The frame layer, with which we perform late fusion, draws from almost all layers, but chooses word-nn marginals over those of the CRF model for the word layer; this shows the effectiveness of the RNN architecture, at least with respect to classification accuracy.

For the initial layers (state, phone, syllable, and word), we observe in the 'ENΔ' column of Table IV that English recovery is best at the state layer and decreases through to the word layer. The is shown as a graph in Figure 8. We believe this is due to the large size of the word segments in comparison with other layers; it easily propagates errors from the ASR 1-bests that the word segment lengths were derived from. This trend can also be seen for the significances.

Although we have put no explicit emphasis on recovering Chinese segments, the improved English accuracy has pulled up the accuracy of Chinese as well. These improvements are minimal, though, most likely because the Chinese support in the acoustic and language models used in ASR is much stronger than that for English. Notably, the trend for Chinese recovery is the opposite of that for English: it is worst at the state layer but best at the word layer. This could be a reflection of the heavy imbalance between Chinese and English words in the corpus.

The fusion (frame) layer clearly outperforms the initial four layers, and additionally shows in Table IV and Figure 8 what looks to be an overtraining trend that peaks at the 4th iteration. This shows (1) that even at the frame level, improvements in CRF classification do not always translate to improvements in



(a) CRF F-measure.

(b) Recognition accuracy.

Fig. 9: GMM-HMM system: Development set performance for English when different proportions of Chinese-only utterances are included in the training corpus. Results shown for frame level, fourth GFS iteration.

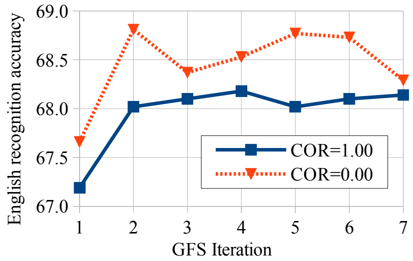


Fig. 10: GMM-HMM system: Development set English recognition accuracies with (COR=1.00) and without (COR=0.00) Chinese-only utterances. Results shown for frame level.

recognition accuracy, and (2) perhaps the addition of the word-nn features has hurt the model. Interestingly, the significances for the fusion layer are considerably higher than those for the initial layers.

Figure 9 shows the results of data-imbalance experiments we conducted on the development set. We generated random subsets of the training corpus by varying the proportion of Chinese-only utterances that were included in the corpus, training a CRF model on that corpus, and evaluating the resultant performance of inference on the development set. All utterances containing English were included in the training corpus subsets. Note the contrast between the rising trend for CRF F-measure and the falling trend for ASR accuracy. Table V and Figure 10 shows the results when we further used the 0.00-ratio training corpus subset to likewise evaluate the performance of all the GFS iterations for the frame layer. Performance is consistantly improved over that in Table IV.

Table VI shows the final recognition results for the test set. Our best result (ratio-0.00 Frame:5) yields a relative improvement of 11.5% over the ASR first-pass baseline, with a significance of 99.9%. Also, it outperforms the MFCC-DNN LID baseline, which uses only frame-level acoustic (MFCC) features. In addition, it outperforms frame GLD, which achieved a relative improvement of 10.1% for English, albeit with stronger results for Chinese. The oracle results, especially those for English, show that there is still much room for improvement in recognition accuracies if better classification can be achieved.

Table VI also shows frame-level experiments conducted on DNN models (DNN Frame). As detailed in Section V-A, DNN models were trained on the same input as the frame-level (late fusion) CRF models described in this section. Similar corpus filtering techniques were used for the DNN models to mitigate the effects of data imbalance. The proposed approach outperforms this DNN frame model as well.

B. *Hybrid system*

The results of GFS for the hybrid system are shown in Tables VII and VIII. These tables are identical to Tables III and IV but reflect the results when the hybrid-generated lattices were used to generate the BPFs, the foundation of the proposed CMLID method. Note that for the hybrid system experiments, we set the Chinese-only ratio (COR) to 0.00 for all layers.

In Table VII we observe higher CRF classification accuracies than those for the GMM-HMM system; this is clearly due to the higher accuracies from the stronger DNN AM, as listed in Table II. In terms of CRF classification accuracy (Table VII), the syllable layer appears to be the strongest, but in terms of ASR accuracies (Table VIII), the syllable layer is clearly the weakest. This shows again that CRF classification accuracy does not necessarily predict ASR accuracy.

We also observe that both state and phone layers selected word-level lattice features (**WD**.`lm_marg2` and **WD**.`am_marg_summary`), and that the word layer in general appears stronger CRF-wise than in the GMM-HMM system. This is likely due to the hybrid system's more accurate lattices from which these features were extracted. ASR-wise, however, we see that the state layer's second iteration was the strongest model of all – even stronger than the fusion layer. This shows that the phone and state layers offer different but complementary cues, but it was also unexpected, as it was thought that the fusion layer would incorporate the most informative cues from the various higher layers to produce the most reasonable decision for every frame. This counter-intuitive result must be attributed again to the mismatch between CRF classification accuracy and ASR accuracy: perhaps the hybrid system's higher first-pass recognition accuracies have intensified this mismatch.

Most notably, the improvements that CMLID brings to the hybrid system are much less than the improvements it brings to the GMM-HMM system. This clearly has much to do with the higher baseline recognition accuracies of the hybrid system. This is likely also because the hybrid system has already captured relevant CMLID cues to some extent; this is similar to the situation described in [37], where the acoustic modeling of standard DNN was found to be much more robust than even highly-specialized, highly-complex, multi-pass GMM-HMM-based approaches. In any case, it is more difficult to improve on the results of a stronger system.

Re-casting these accuracy improvements as error rate reductions sheds a slightly different light on the matter: under the GMM-HMM framework, using the proposed CMLID method we reduced the error rate by 17% relative (37.6% to 31.2%). To maintain this rate of improvement on the hybrid system using CMLID would clearly require a considerably greater effort; thus a 5.3% reduction of this strong baseline's error rate (20.6% to 19.5%) is an accompishment indeed, especially

TABLE V: *GMM-HMM system: Data imbalance results for final frame layer: development set recognition accuracies and CRF F-measures. Chinese-only ratio set to* 0.00.

GFS Iteration	# fea	CH	EN	OV	Sig	CHΔ	ENΔ	OVΔ	SIL	CH	EN
1	131600	83.4	67.7	82.2	99.1	0.6	8.4	1.1	0.628	0.843	0.680
2	157700	83.5	68.8	82.3	99.6	0.7	10.3	1.3	0.627	0.841	0.687
3	168969	83.5	68.4	82.3	99.5	0.7	9.6	1.2	0.632	0.842	0.692
4	179256	83.5	68.5	82.3	99.5	0.7	9.8	1.3	0.635	0.843	0.697
5	**183187**	**83.5**	**68.8**	**82.3**	**99.6**	**0.7**	**10.2**	**1.2**	0.636	0.841	0.696
6	178370	83.5	68.7	82.3	99.6	0.7	10.2	1.2	0.637	0.842	0.698
7	190111	83.5	68.3	82.3	99.1	0.7	9.5	1.2	0.641	0.844	0.700

TABLE VI: *GMM-HMM system: Final test set recognition accuracies. Frame GLD is the result from [6]. COR = Chinese-only ratio.*

Experiment	GFS Iteration	COR	CH	EN	OV	Sig	CHΔ	ENΔ	OVΔ
Baseline			83.4	60.4	81.6				
MFCC-DNN LID		0.00	83.8	65.3	82.4	99.0	0.6	8.1	1.0
Frame GLD			84.1	66.5	82.8	99.9	0.9	10.1	1.4
CRF State	3	1.00	84.1	66.6	82.8	99.4	0.9	10.3	1.4
CRF Phone	3	1.00	84.0	66.1	82.7	99.6	0.8	9.5	1.3
CRF Syllable	3	1.00	84.0	66.3	82.7	99.6	0.8	9.8	1.3
CRF Word	3	1.00	84.0	65.5	82.6	99.2	0.8	8.4	1.2
CRF Frame	4	1.00	84.0	66.8	82.7	99.9	0.8	10.7	1.3
CRF Frame	**5**	**0.00**	**84.0**	**67.3**	**82.7**	**99.9**	**0.7**	**11.5**	**1.3**
DNN Frame		0.00	84.2	66.3	82.9	99.6	1.0	9.8	1.6
Oracle			84.8	72.1	83.9	100.0	1.8	19.4	2.7

TABLE VII: *Hybrid system: GFS feature group ensembles, feature function counts, and development set CRF F-measures.*

Layer	GFS iteration	Feature groups (cumulative)	# fea	SIL	CH	EN
State	1	⟨phoneme1+**WD**.lm_marg2⟩	20890	0.446	0.877	0.741
	2	+ ⟨phoneme1+phoneme2⟩	56234	0.468	0.880	0.760
	3	+ ⟨phoneme2+**SYL**.lm_marg_summary⟩	69725	0.468	0.881	0.768
Phone	1	phoneme1	20765	0.464	0.905	0.774
	2	+ ⟨phoneme2+**ST**.marg_summary⟩	44833	0.464	0.907	0.791
	3	+ ⟨lap1+**WD**.am_marg_summary⟩	49675	0.475	0.909	0.800
Syllable	1	⟨syl.orig+syl.cv⟩	19876	0.648	0.903	0.782
	2	+ ⟨rap1.coda+**PH**.marg_summary⟩	24059	0.655	0.907	0.803
	3	+ ⟨pitch_slopebegin+**PH**.wc⟩	27724	0.681	0.919	0.817
Word	1	⟨**PH**.wc+**SYL**.marg_summary⟩	12082	0.867	0.883	0.617
	2	+ ⟨word+left.energy_max⟩	21446	0.872	0.890	0.638
	3	+ ⟨**WD**.am_marg_summary+**SYL**.awl⟩	24021	0.881	0.894	0.649
Frame	1	⟨phoneme1+**SYL**.marg_summary⟩	120956	0.631	0.859	0.753
	2	+ **ST**.marg2	120616	0.631	0.859	0.757
	3	+ phoneme1	133549	0.636	0.861	0.758
	4	+ ⟨**ST**.marg1+**PH**.awl⟩	140079	0.642	0.863	0.760
	5	+ ⟨**SYL**.marg_summary+**WD**.awl⟩	152046	0.641	0.863	0.762
	6	+ ⟨phoneme2+**SYL**.awl⟩	164459	0.642	0.863	0.763
	7	+ ⟨**SYL**.awl+**SYL**.marg2⟩	167354	0.643	0.863	0.764

considering the limited data (only 9 hours, of which we use only 4.5h) available. Indeed, extensive further tuning of the various hybrid AM's parameters yielded only negligible improvements in recognition accuracy, far less than those gained with the CMLID method.

These observations hold also with Table IX, which shows the final test set recognition accuracies for the hybrid system. Note in particular the oracle results, which further indicate the limited potential for improvement for this strong system.

C. Discussion

Why do the word and word-nn layers yield such small improvements in recognition accuracy for the GMM-HMM system? We believe this is due to the longer tokens in the word layer, which propagate errors more. One potential solution to this problem would be to take into account hypotheses other than only the 1-best hypothesis, for instance using the lattice or derived N-best lists. Otherwise it is indeed difficult when using such a framework to avoid promoting the original system hypothesis. Nevertheless, the lower-level layers clearly were able to recover information lost or obscured at the word level. For instance, the frame layer's choice in Table III of feature conjunction ⟨conf.lat1+**WD_NN**.marg0⟩ indicates that despite the poor performance of the word-nn layer, when considered in conjunction with the average confidence of the left few frames, word-nn marginals aid in language classification. The word layer for the hybrid system was stronger than that in the GMM-HMM system, most likely because of the more-accurate base lattices.

Which model is better for the LID task described here: CRF or DNN? DNN's inherent multi-layer topology makes it a more elegant solution, because it thus has the potential to capture high-level feature interactions without explicit user intervention. The GFS algorithm described here, however, is a feature engineering method to explicitly find useful high-level features for CRF models. The results in Table VI show that for the highest performance yields, it is still necessary to utilize methods like GFS for CRF models.

TABLE VIII: *Hybrid system: Development set recognition accuracies.*

	GFS	ASR accuracy				CH	EN	OV
Layer	iter.	CH	EN	OV	Sig	Δ	Δ	Δ
Baseline		87.5	79.4	86.8				
State	1	87.6	80.4	87.0	77.9	0.2	1.2	0.2
	2	**87.6**	**80.5**	**87.0**	**80.6**	**0.1**	**1.4**	**0.2**
	3	87.6	80.3	87.0	76.8	0.1	1.1	0.2
Phone	1	87.6	80.2	87.0	80.7	0.2	1.0	0.2
	2	87.6	80.4	87.0	81.3	0.1	1.2	0.2
	3	87.6	80.4	87.0	76.1	0.1	1.2	0.2
Syllable	1	87.6	80.0	87.1	75.2	0.2	0.8	0.2
	2	87.6	80.0	87.0	73.5	0.2	0.7	0.2
	3	87.6	79.9	87.0	73.0	0.1	0.6	0.2
Word	1	87.6	80.0	87.0	75.0	0.2	0.7	0.2
	2	87.6	80.2	87.1	77.3	0.2	1.0	0.3
	3	87.6	80.4	87.1	79.2	0.2	1.3	0.2
Frame	1	87.6	80.2	87.0	73.3	0.2	0.9	0.2
	2	87.6	80.2	87.0	74.4	0.1	1.0	0.2
	3	87.6	80.0	87.0	70.0	0.1	0.7	0.2
	4	87.6	80.1	87.0	73.4	0.1	0.8	0.2
	5	87.6	80.2	87.0	77.4	0.1	1.0	0.2
	6	87.6	80.2	87.0	75.9	0.1	1.0	0.2
	7	87.6	80.1	87.0	73.1	0.2	0.8	0.2

In a CRF-centric paradigm, GFS works well because CRF models with a small number of features train quickly. As such, the concatenating of atomic features (that is, the feature conjunctions of Section IV-A3) is a feasible way to capture high-level information. With DNN models such as Table VI's DNN Frame, however, such feature conjunctions result in large feature counts, which yield large input layers and correspondingly large models, which can take weeks to train, even with high-end GPUs. Since it was not immediately apparent how to efficiently incorporate feature conjunctions into a DNN framework, we elected instead to extend the DNN topology by increasing the number of internal layers (to five) in the hopes that so doing would achieve the same goal of capturing high-level information but in an even more convenient way than GFS. However, additional DNN depth (that is, internal layers) does not capture temporally contextual cues. We attempted to extend temporal context by widening the input layer to include the features of more surrounding frames (10 previous + 10 following), but this yielded no additional improvements.

A closer look at error patterns present in the proposed and oracle results (Table X) as compared to the GMM-HMM system's baseline and frame GLD results shows that the improvements yielded are largely attributable to reductions in substitution errors. This is clearly due to the increased frame-level accuracy in language identification, which in certain cases helps the recognizer to "hear" word segments more clearly. That is, by boosting the likelihoods for the detected language's state models, it effectively budges the recognizer in what is judged to be the right direction. This is, of course, the purpose of the proposed method.

VIII. Conclusion

We have proposed a complete framework for the recovery of English segments in code-mixed speech recognition. The strength of this approach lies in the combination of both low- and high-level features, and the selection of the most effective of these features, to yield a small set of nonlinear, high-order features highly targeted to English-specific code-mixed language identification. Our use of k-way cross-validation with acoustic models and CRFs when extracting cascaded features yields improved performance, as does tuning the data imbalance ratio between Chinese and English: this is important, since code-mixed tasks often involve highly imbalanced data. We have proposed a simple and exact but highly parallelizable method for the induction of CRF feature groups, and demonstrated its effectiveness on a code-mixed Chinese-English lecture corpus. Together, these techniques are shown to yield improved performance over previous work, even over high-performance new methodologies for LID like DNN, or new hybrid recognition paradigms such as CD-HMM-DNN, and represent a foundation for future development in the recovery of guest-language segments in limited-resource code-mixed speech.

In the future, for GMM-HMM systems we hope to generate stronger features, for instance articulatory features that are classifier-based instead of knowledge-based. We would also like to perform GFS at the feature rule level instead of the feature group level, and use wider contexts and investigate using 3-way feature conjunctions to capture higher-order information. Additionally, further work on neural network-based methods such as DNNs could result in performance breakthroughs; in particular, one promising direction is the use of an RNN- or LSTM-based [38], [39] framework which would allow for stronger implicit temporally contextual cues in a neural network.

For hybrid systems, the focus of future work should be on exploring more fully the potential of deep learning models to accomplish implicit CMLID. This could be as simple as adding depth or width to the model, or it could involve multi-task type structural hints to encourage the model to "pay more attention" to languages. Other directions include developing new DNN input features, for instance prosodic features (pitch/energy/duration) based on auto-detected pseudo-syllables. Of course, additional context is always helpful but may be difficult to model with such limited data.

References

[1] S. Sridhar and K. K. Sridhar, "The Syntax and Psycholinguistics of Bilingual Code-mixing," *Canadian Journal of Psychology*, 1980.

[2] P. Li, "Spoken Word Recognition of Code-switched Words by Chinese-English Bilinguals," *Journal of Memory and Language*, 1996.

[3] F. Grosjean, *Bilingual: Life and Reality*. Harvard University Press, 2010.

[4] "United States Census Bureau, American Community Survey," http://www.census.gov/acs/www/.

[5] C.-F. Yeh, L.-C. Sun, C.-Y. Huang, and L.-S. Lee, "Bilingual Acoustic Modeling with State Mapping and Three-stage Adaptation for Transcribing Unbalanced Code-mixed Lectures," in *Proceedings of ICASSP*, 2011.

[6] C.-F. Yeh, A. Heidel, H.-Y. Lee, and L.-S. Lee, "Recognition of Highly Imbalanced Code-Mixed Bilingual Speech with Frame-level Language Detection Based on Blurred Posteriorgram," in *Proceedings of ICASSP*, 2012.

[7] F. Grosjean, "Exploring the recognition of guest words in bilingual speech," *Language and Cognitive Processes*, vol. 3, 1988.

[8] C. Soares and F. Grosjean, "Bilinguals in a monolingual and a bilingual speech mode: The effect on lexical access," *Memory and Cognition*, vol. 12, 1984.

TABLE IX: *Hybrid system: Final test set recognition accuracies. COR = Chinese-only ratio.*

Experiment	GFS Iteration	COR	CH	EN	OV	Sig	CHΔ	ENΔ	OVΔ
Baseline			87.6	76.9	86.8				
CRF State	2	0.00	87.8	78.3	87.1	80.2	0.2	1.8	0.3
Oracle			87.9	80.6	87.4	99.1	0.4	4.8	0.6

TABLE X: *GMM-HMM system: Sample errors that are corrected in proposed approach and oracle experiments.*

Frame GLD	Proposed method
X 也是 (*yeshi*)	ACCURACY
L 的也 (*deye*)	LDA
LAMBDA 去(*qu*) INDEPEN 的(*de*)	LANGUAGE INDEPENDENT
FOR THE 放(*fang*)	FALSE ALARM
FOR 什麼(*sheme*)	FALSE ALARM

Proposed method	Oracle experiment
比 (*bi*)	B
CONTINUOUS	CONTINUOUSLY
ON 得到 (*dedao*)	ORTHOGONAL
IS PERIMETER FRONT TRAIN	DISCRIMINATIVE TRAINING
P K SHIFT	APPLICATION
N TWO 這邊的 (*zhebiande*)	ENTROPY 變得 (*biande*)

[9] D. Foss, "Decision processes during sentence comprehension: Effects of lexical item difficulty and position upon decision times," *Journal of Verbal Learning and Verbal Behavior*, vol. 8, 1969.

[10] F. Grosjean, "The recognition of words after their acoustic offset: Evidence and implications," *Perception and Psychophysics*, vol. 38, 1985.

[11] J. McClelland and J. Elman, "The TRACE model of speech perception," *Cognitive Psychology*, vol. 18, 1986.

[12] M. A. Zissman, "Comparison of Four Approaches to Automatic Language Identification of Telephone Speech," *IEEE Transactions on Speech and Audio Processing*, 1996.

[13] P. A. Torres-carrasquillo, E. Singer, M. A. Kohler, and J. R. Deller, "Approaches to Language Identification Using Gaussian Mixture Models and Shifted Delta Cepstral Features," in *Proceedings of ICSLP*, 2002.

[14] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-carrasquillo, "Support Vector Machines for Speaker and Language Recognition," *Computer Speech & Language*, 2006.

[15] Y. C. Chan, P. C. Ching, T. Lee, and H. M. Meng, "Detection of Language Boundary in Code-switching Utterances By Bi-phone Probabilities," in *International Symposium on Chinese Spoken Language Processing*, 2004.

[16] Y. C. Chan, P. C. Ching, T. Lee, and H. Cao, "Automatic Speech Recognition of Cantonese-English Code-mixing Utterances," in *Proceedings of ICSLP*, 2006.

[17] H. Cao, P. C. Ching, and T. Lee, "Effects of Language Mixing for Automatic Recognition of Cantonese-English Code-Mixing Utterances," in *Proceedings of Interspeech*, 2009.

[18] D. Imseng, H. Boulard, M. Magimai-Doss, and J. Dines, "Language dependent universal phoneme posterior estimation for mixed language speech recognition," in *Proceedings of ICASSP*, 2011.

[19] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *International Conference on Machine Learning*, 2001.

[20] C. Sutton and A. McCallum, *An Introduction to Conditional Random Fields for Relational Learning.* MIT Press, 2006, ch. 4.

[21] T. Mikolov, M. Karafiat, J. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of Interspeech*, 2010.

[22] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *IEEE Spoken Language Technology Workshop*, 2012.

[23] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proceedings of ICASSP*, 2011.

[24] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Cernocky, "Strategies for training large scale neural network language models," in *IEEE Automatic Speech Recognition and Understanding Workshop*, 2011.

[25] A. McCallum, "Efficiently inducing features of conditional random fields," in *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, 2003.

[26] T. Kudo, http://chasen.org/~taku/software/CRF++, 2005–2013.

[27] C.-A. Chan and L.-S. Lee, "Unsupervised spoken-term detection with spoken queries using segment-based dynamic time warping," in *Proceedings of Interspeech*, 2010.

[28] B. Yin, E. Ambikairajah, and F. Chen, "Voiced/Unvoiced Pattern-based Duration Modeling for Language Identification," in *Proceedings of ICASSP*, 2009.

[29] C.-K. Lin and L.-S. Lee, "Improved features and models for detecting edit disfluencies in transcribing spontaneous Mandarin speech," *IEEE Transactions on Audio, Speech, and Language Processing*, 2009.

[30] J. Fayolle, F. Moreau, C. Raymond, G. Gravier, and P. Gros, "CRF-based Combination of Contextual Features to Improve A Posteriori Word-level Confidence Measures," in *Proceedings of Interspeech*, 2010.

[31] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[32] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[33] P.-W. Chou, https://github.com/botonchou/libdnn.

[34] C.-F. Yeh, C.-Y. Huang, and L.-S. Lee, "Bilingual acoustic model adaptation by unit merging on different levels and cross-level integration," in *Proceedings of Interspeech*, 2011.

[35] L. F. Chien, "PAT-tree-based adaptive keyphrase extraction for intelligent Chinese information retrieval," in *Proceedings of the 20th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, 1997.

[36] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE Automatic Speech Recognition and Understanding Workshop*, 2011.

[37] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proceedings of ICASSP*. IEEE, 2013, pp. 7398–7402.

[38] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[39] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proceedings of Interspeech*, 2012.

Aaron Heidel received the B.S. and M.S. degrees in computer science and information engineering from National Taiwan University, Taipei, Taiwan in 2005 and 2007, respectively. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan. His research is focused on automatic speech recognition.

Hsiang-hung Lu received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan in 2014 and is currently pursuing the M.S. degree in the Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan. His research is focused on acoustic modeling and automatic speech recognition.

Lin-shan Lee (F3) received the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA. He has been a Professor of electrical engineering and computer science at the National Taiwan University, Taipei, Taiwan, since 1982 and holds a joint appointment as a Research Fellow of the Academia Sinica, Taipei. His research interests include digital communications and spoken language processing. He developed several of the earliest versions of Chinese spoken language processing systems in the world, including text-to-speech systems, natural language analyzers, dictation systems, and voice information retrieval systems. Dr. Lee was Vice President for International Affairs (1996–1997) and the Awards Committee chair (1998–1999) of the IEEE Communications Society. He was a member of the Board of ISCA (International Speech Communication Association) (2002–2009), a Distinguished Lecturer (2007–2008) of the IEEE Signal Processing Society, and the general chair of ICASSP 2009 in Taipei. He has been a fellow of ISCA since 2010, and received the Meritorious Service Award from the IEEE Signal Processing Society in 2011.