A Real-Time End-to-End Multilingual Speech Recognition Architecture

Javier Gonzalez-Dominguez, *Member, IEEE*, David Eustis, Ignacio Lopez-Moreno, *Member, IEEE*, Andrew Senior, *Senior Member, IEEE*, Françoise Beaufays, *Senior Member, IEEE*, and Pedro J. Moreno, *Senior Member, IEEE*

***Abstract*—Automatic speech recognition (ASR) systems are used daily by millions of people worldwide to dictate messages, control devices, initiate searches or to facilitate data input in small devices. The user experience in these scenarios depends on the quality of the speech transcriptions and on the responsiveness of the system. For multilingual users, a further obstacle to natural interaction is the monolingual character of many ASR systems, in which users are constrained to a single preset language. In this work, we present an end-to-end multi-language ASR architecture, developed and deployed at Google, that allows users to select arbitrary combinations of spoken languages. We leverage recent advances in language identification and a novel method of real-time language selection to achieve similar recognition accuracy and nearly-identical latency characteristics as a monolingual system.**

***Index Terms*—Automatic speech recognition (ASR), deep neural network (DNN), language identification (LID), multilingual.**

I. Introduction

AUTOMATIC speech recognition (ASR) has become increasingly relevant to date, tracking the explosive growth of mobile devices. The use of voice as a natural and convenient method of human-device interaction is especially applicable to hands-free scenarios (e.g., while driving) and interaction with small form-factor devices (e.g., wearables). The quality of the user experience in these scenarios is primarily affected by the transcription accuracy and real-time responsiveness of the ASR system.

For multilingual users, another obstacle to natural interaction is the common monolingual character of ASR systems, in which users can speak in only a single preset language. According to several sources [1]–[3], multilingual speakers already outnumber monolingual speakers, and predictions point to a larger number of multilingual speakers in the future. The capacity to transparently recognize multiple spoken languages is, therefore, a desirable feature of ASR systems.

Manuscript received July 01, 2014; accepted October 01, 2014. Date of publication October 23, 2014; date of current version May 12, 2015. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Raphael Cendrillon.

J. Gonzalez-Dominguez is with Google, Inc., New York, NY 10011 USA, and also with the ATVS Biometric Recognition Group, Universidad Autonoma de Madrid, 28046 Madrid, Spain (e-mail: jgd@google.com; javier.gonzalez@uam.es).

D. Eustis, I. Lopez-Moreno, A. Senior, F. Beaufays, and P. J. Moreno are with Google, Inc, New York, NY 10011 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTSP.2014.2364559

Several architectures have been considered to achieve multilingual speech recognition. One technique has been to train a universal speech model capable of recognizing multiple languages. Efforts in this direction are presented in [4]–[6]. This approach seeks to exploit similarities among languages and dialects, and lends itself to an easily deployable system. However, universal models tend to be larger and higher in perplexity relative to their monolingual equivalents, leading to potentially adverse effects on transcription accuracy and decoding latency.

Other architectures have attempted to detect the language of an utterance as a preprocessing step, through the use of language identification (LID) classifiers [7] [8]. Here, the outcome of the LID classification determines which of several monolingual speech recognizers is activated. The main drawbacks of this method are the latency introduced by the LID step, and the propagation of language classification errors to the final transcription.

Here, we present an integrated end-to-end multilingual architecture that builds upon the work described in [9]. In this architecture, monolingual speech recognizers decode the input signal simultaneously in each of the selected languages, while the LID system tries to determine which language is spoken. A decision is then made, based on the LID decision and on the confidence scores of the individual recognizers as to which recognition result best matches the user input. This architecture avoids the extra latency that an early language decision would introduce, and benefits from the extra scores from the recognizers to better decide which result to return to the user. These benefits come however with an increased processing cost since the input is recognized multiple times.

In this paper, we further improve upon [9] in two major directions. First, we replace the LID classifier with a more accurate Deep Neural Network (DNN) based classifier. Second, we propose a method to dynamically combine confidence scores and LID decisions from partial recognition results together with various timeout strategies to maintain the streaming nature of the monolingual system and to greatly reduce computing costs. One such strategy is shown to perform with near-ideal characteristics in both dimensions, minimizing latency and misrecognitions caused by language confusions.

Both the ASR and LID backend engines are based on DNNs [10]–[12]. During recent years, DNNs have achieved outstanding performance in diverse and challenging machine learning applications. Those including acoustic modelling [13] [14], visual object recognition [15], and many others [16]. Compared to previous acoustic modelling based on GMMs, the

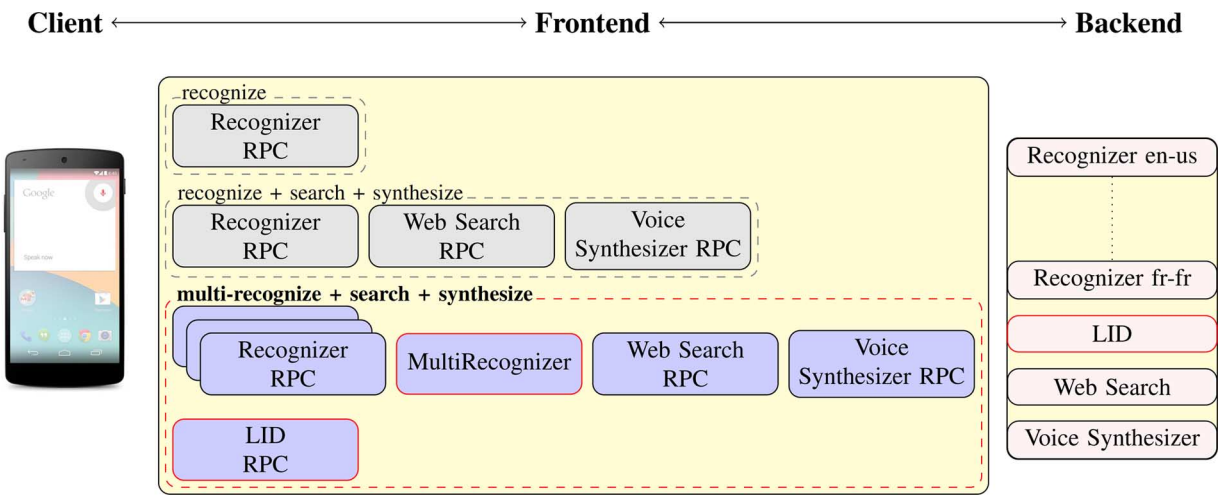

Fig. 1. Diagram showing components of the complete architecture: Client, Frontend and Backends. Within the Frontend, there are several activity pipelines composed of modules, each of which communicates with a single Backend. In the "multi-recognize + search + synthesize" activity pipeline, we introduce a MultiRecognizer module responsible for language selection.

use of DNNs presents several advantages. First, unlike GMMs, DNNs employ a multilevel distributed representation of the input [14]. This fact makes DNNs exponentially more compact than GMMs [17]. Second, DNNs, being purely discriminative, do not impose any assumptions about the input data distribution. Further, DNNs have proved successful in exploiting large amounts of data, achieving more robust models without lapsing into overfitting. We leverage this last point to employ large quantities of training data recorded from user traffic.

The rest of this paper is organized as follows. Section II presents the overall architecture of the system. Section III describes the frontend component which is responsible for language selection. Section IV describes the backend components, with Section IV-A focusing on the LID classifier and Section IV-B focusing on the monolingual speech recognizers. In Section V we present the databases and evaluation metrics used to assess system performance. We present results in Section VI. Finally, conclusions are outlined in Section VII.

II. Overall Architecture

The end-to-end multilingual speech recognition system consists of the following components represented in Fig. 1:

- ***Client*** : a mobile phone, browser, or similar internet connected device capable of recording audio, issuing voice requests and displaying results.
- ***Frontend***: an HTTP server exposing a voice recognition API, which serves as an intermediary between the client and the backend(s). The frontend supports several activity pipelines. For example, one activity called "recognize" simply converts an audio request to a text transcription. A more complex activity pipeline called "recognize + search + synthesize" executes a web search based on the transcription, and then synthesizes an audio summary for the user. Frontend activity pipelines are composed of modules, each of which performs some task such as communicating with a backend over RPC. In the multi-language configuration, we add an additional MultiRecognizer module which is responsible for making language selection decisions.
- ***Backends***:
  - **LID Backend**: a Remote Procedure Call (RPC) server that accepts an audio stream and returns language identification scores for many languages, as described in Section IV-A.
  - **Speech Recognizer Backend**: hereafter Recognizer, an RPC server that accepts an audio stream and streams back transcription results for a particular language (e.g., US English), as described in Section IV-B .
  - **Web Search Backend**: an RPC server that accepts a recognition transcription and retrieves search results. The results sometimes include a text summary which is intended to be read back to the user. Search results are also determined by the selected language, which affects the search corpus, query normalization procedure, and result summarization procedure.
  - **Voice Synthesizer Backend**: an RPC server that accepts text (in this case the search summary), and produces a speech waveform. Again, voice synthesis is influenced by the selected language. For example, depending on which of two dialects is selected, the speech output will be accented toward that dialect.

All components communicate via bi-directional streams. The Client begins sending audio segments to the Frontend as soon an utterance begins. The Frontend forwards those segments to a Recognizer to obtain partial transcriptions results, which are then returned to the Client for display. Typically, the Client receives several partial transcriptions before the user finishes speaking an utterance, followed by a final transcription. For the purpose of display, each subsequent transcription overrides the previous, providing real-time feedback to the user.

In the single-language voice recognition system, the Client provides one spoken language preference which directs the Frontend to the corresponding Recognizer. In the multi-language system, the client may provide several language candidates, causing the Frontend to forward the audio stream to multiple Recognizers in parallel as well as to the LID

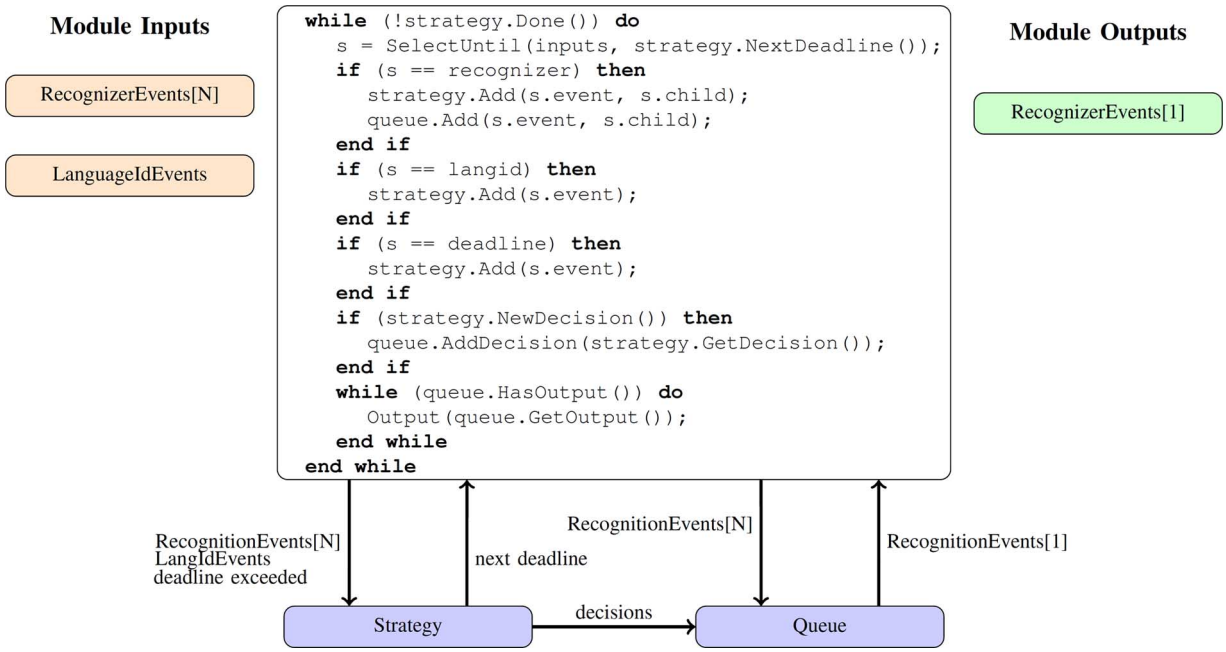

Fig. 2. Pseudocode illustrating how the MultiRecognizer accepts input streams from the LID classifier and monolingual recognizers, and emits an output stream emulating a universal recognizer.

Backend. The Frontend then makes real-time language selection decisions based on all of these backend results and some strategy. Only transcription results from the selected language are returned to the Client, thus emulating a universal speech recognizer. In the event that the language selection decision changes mid-stream, the Frontend emits enqueued transcription responses for the new selected language, causing the Client to display the transcription for the new language.

III. Multilingual Recognizer Frontend

A. MultiRecognizer Module

The MultiRecognizer module encapsulates the language selection logic in the Frontend. It accepts as inputs i) a stream of transcription results from multiple Recognizers, where each transcription includes a confidence score indicating the probability that the transcription is correct and ii) a stream of language classification scores from the LID Backend. Its output is a single stream of transcription results. As the MultiRecognizer module receives transcription results from the Recognizers it adds them to a queue. This queue withholds results until appropriate language selection decisions have been made. A Strategy class examines all Recognizer and LID input events and outputs language selection decisions. The decisions are designated either partial or final, and cause the queue to release partial or final transcription results for the selected language. Released results are then returned to the client. Pseudocode for this process is depicted in Fig. 2.

The Strategy class must decide which language to select, and also importantly, *when* to emit its decisions. It will generally make more accurate decisions as more information becomes available. However, delaying a decision may introduce latency, particularly because Recognizers typically respond more slowly when decoding audio from an unexpected language. Note that latency is incurred precisely when the correct Recognizer has finished, but the Strategy is waiting for results from incorrect ones. In principle, latency could also be incurred waiting for classification scores from the LID Backend, but in practice this typically responds faster than any Recognizer, and furthermore can be tuned to provide frequent partial responses (e.g., every 200 ms) which are cumulatively averaged.

Formally, after an input event at time t, the Strategy class updates partial scores for language i, $s_{i,t}$, as a combination of the speech transcription confidence and the LID Backend output probability for that language. In this work, we use linear weighting to combine these as

$$s_{i,t} = \alpha p(\mathcal{W}i|L_i, t_1, \theta_{ASR_i}) + \beta \frac{1}{T_2} \sum_{t=0}^{t_2} p(L_i|t_2, \theta_{LID}) \quad (1)$$

where,

$p(W_i|L_i, t_1, \theta_{ASR_i})$ is the confidence of the last transcription received at time $t_1 \leqq t$ using the Recognizer model for language i, θ_{ASR_i}.

$p(L_i|t_2, \theta_{LID})$ is the language identification output probability for language i, at time $t_2 \leqq t$, produced for the LID model θ_{LID}.

and α and β are adjustable parameters to weight the influence of the Recognizers and the LID system.

Note that one of t_1 or t_2 will be equal to t depending on whether the input event is produced by a Recognizer or by the LID Backend. If the input event was produced by the LID Backend, we do not compute $s_{i,t}$ for languages, i, where the corresponding Recognizer has not yet returned any transcription. The final score for a given language, s_{iT_f}, is also updated as in (1) when a final response for the corresponding Recognizer is received $(t = T_f)$.

After the update step, the Strategy class tests for certain decision triggers. In this work we consider three timeout strategies for the final decision trigger. Those are:

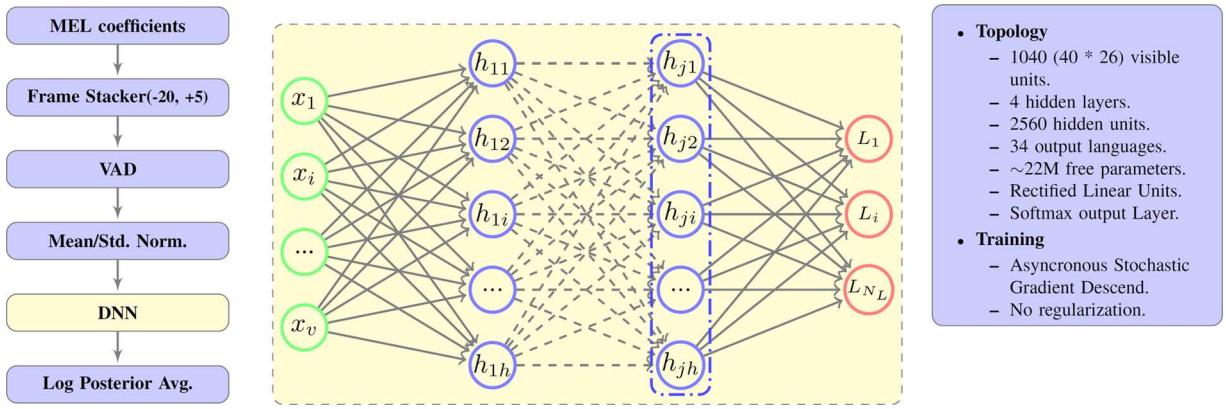

Fig. 3. Pipeline process from the waveform to the final score (left). DNN topology (middle). DNN description (right).

- *Infinite timeout strategy*: always wait for the final response from all Recognizers.
- *Constant timeout strategy*: wait for the first final response from any Recognizer, then timeout after some constant duration (τ_{cte}). Setting τ_{cte} to 1s provided the best experimental results.
- *Variable timeout strategy*: modify the constant timeout duration by examining the difference between the highest available final score and the highest available partial score from a Recognizer that we are still waiting on. Let I be the set of languages that have already received a final Recognizer response and let J be the remaining languages that we are still waiting on; formally we set the variable timeout τ as:

$$\tau = \tau_{cte} \max(1 - \gamma s_\delta, 0) \tag{2}$$

with

$$s_\delta = \max_I(s_{i,T_{fi}}) - \max_J(s_{j,t}). \tag{3}$$

Thus, when s_δ is positive, the timeout duration (τ) is shorter (or zero). On the other hand when s_δ is negative, τ is longer. We tune γ to control the variability of the timeout. Note that if either s_δ or γ is zero, the equation degenerates to the Constant timeout strategy. Setting τ_{cte} to 1s and γ to 2.0 provided the best experimental results.

Given one of the above timeout strategies, the Strategy class uses the following triggers for emitting partial or final language decisions:

- *Partial language decision*: if there exists a previous partial language decision, l, then a new partial decision is triggered when $\arg\max_i(s_{it}) \neq l$. Otherwise, if l does not yet exist, then the trigger is $\max(s_{it}) > 0$.
- *Final language decision*: a final decision is produced when either there is a final response from all the Recognizers ($s_{i,T_{fi}} > 0, \forall i$) or the timeout from the given timeout strategy is exceed ($t > \tau$).

IV. Backends

We describe in this section the language identification and monolingual speech recognition backend engines.

A. Deep Neural Networks for LID

1) Architecture: The DNN used for LID is a fully-connected feed-forward neural network with hidden units implemented as rectified linear units (ReLU). Thus, an input at level j, x_j, is mapped to its corresponding activation y_j (input of the layer above) as

$$y_j = ReLU(x_j) = \max(0, x_j) \tag{4}$$

$$x_j = b_j + \sum_i w_{ij} y_i \tag{5}$$

where i is an index over the units of the layer below and b_j is the bias of the unit j.

The output layer is then configured as a *softmax*, where hidden units map input x_j to a class probability p_j in the form

$$p_j = \frac{\exp(x_j)}{\sum_l \exp(x_l)} \tag{6}$$

where l is an index over all the classes.

As a cost function for backpropagating gradients in the training stage, we use the cross-entropy function defined as

$$C = -\sum_j t_j \log p_j \tag{7}$$

where t_j represents the target probability of the class j for the current evaluated example, taking a value of either 1 (true class) or 0 (false class).

2) Implementing DNN for LID: From the conceptual architecture explained above, we construct a language identification system to work at the frame level as follows.

As the input of the net we used 40 mel filterbanks. Specifically, the input layer was fed with 26 frames formed by stacking the current processed frame and its $-20, +5$ left/right neighbors. Thus, the input layer comprised a total number of $1040(26 \times 40)$ visible units, v. The reason behind using an asymmetric context is to reduce the undesirable latency on processing every frame in a real-time scenario. Note that just 5 frames in the future are required (~ 50 ms).

On top of the input layer, we stacked a total number of N_{hl} (4) hidden layers, each containing h (2560) units. Then, we added the softmax layer, whose dimension (s) corresponds to the number of target languages (N_L).

Overall, the net was defined by a total of w free parameters (weights + bias), $w = (v+1)h + (N_{hl}-1)(h+1)h + (h+1)s$. The complete topology of the network is depicted in Fig. 3.

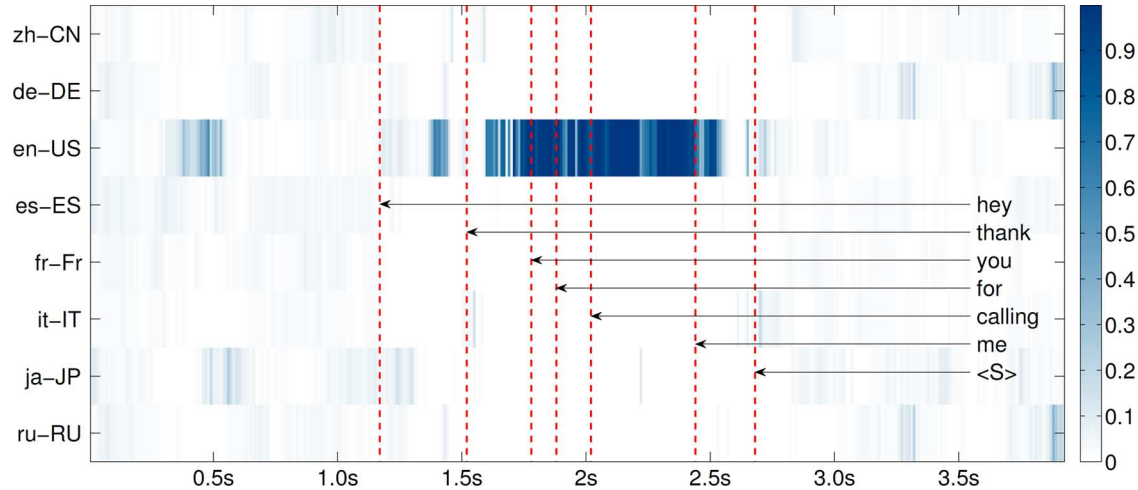

Fig. 4. Color map of frame by frame DNN-based LID system output probabilities (8 languages selected) for an English-USA (4s) test utterance.

Regarding the training procedure, we used asynchronous stochastic gradient descent within the DistBelief framework [10], a software framework that uses computing clusters with thousands of machines. This distributed allows us to exploit large amounts of data to train large models. The learning rate and minibatch size were fixed to 0.001 and 200 samples [1].

Note that the presented architecture works at the frame level, meaning that each single frame (plus its corresponding context) is fed-forward through the network, obtaining a class posterior probability for all of the target languages. This fact makes DNNs particularly suitable for real-time applications since, unlike other approaches (i.e., i-vectors), we can potentially make a decision about the language at each new frame. Indeed, at each frame, we can combine the evidence from past frames to get a single similarity score between the test utterance and the target languages. A simple way of doing this combination is to assume that frames are independent and multiply the posterior estimates of the last layer. The score s_l for the language l of a given test utterance is computed by multiplying the output probabilities p_l obtained for all its frames; or equivalently, accumulating the logs as

$$s_l = \frac{1}{N}\sum_{t=1}^{N} logp(L_l|x_t,\theta) \tag{8}$$

where $p(L_l|x_t,\theta)$ represents the class probability output for the language l corresponding to the input example at time t, x_t by using the DNN defined by parameters θ.

Fig. 4 shows the frame-by-frame DNN outputs (log probabilities $\log p(L_l|x_t,\theta)$), produced for an American English test utterance of 4s duration[2]. The color intensity of each cell (frame/language) represents the output probability of the given frame to belong to the corresponding language.

B. Speech Recognizers

For ASR, we use the same features and similar neural network architectures as are described in Section IV-A for language identification. For each language, we use exactly the same 26-frame asymmetric window of normalized 40-dimensional filterbank energy features, feeding into a network with eight hidden layers of 2560 ReLUs each. The final layer is again a softmax, but here trained to estimate posteriors for 14,000 context dependent triphone states. Such a network has 85 million trainable parameters. The network is trained with asynchronous gradient descent, first to a cross-entropy criterion and then with sequence discriminative training. [18]. Each language's neural network is trained on thousands of hours of speech, using anonymized live traffic utterances. For some languages these are manually-transcribed and in others they are machine transcribed using an earlier generation recognizer. Each language uses a pruned 5-gram language model with on-the-fly rescoring using a much larger 5-gram language model. Numerous optimizations are made to enable decoding in real time on conventional hardware [19].

[1]We define *sample* as the input of the DNN: the feature representation of a single frame besides those from its adjacent frames forming the context.

[2]For the sake of clarity, just 8 out of 34 outputs (languages) are showed.

V. Databases and Evaluation Metrics

1) Google 5M Language Identification Corpus: We generated the Google 5M LID corpus dataset by randomly picking anonymized queries from several Google speech recognition services such as Voice Search or the Speech Android API. Following the user's phone Voice Search language settings, we labelled a total of $\sim$ 5 million utterances, 150 k utterances by 34 different locales (25 languages+9 dialects) yielding $\sim$ 87,5 h of speech per language and a total of $\sim$ 2975 h. A held-out test set of 1k utterances per language was created while the remainder was used for training and development. Involved languages and data description is presented in Figs. 5 and 6.

Non-speech queries were discarded. Selected queries ranged from 1s up to 10s in duration with average speech content of 2.1s. Fig. 6 shows the duration distribution before and after doing this activity detection process.

2) Google Multilang Corpus: In order to constrain the combinatorial explosion of language tuples to be evaluated, we selected 8 languages for the Google Multilang Speech Recognition Corpus. This list is given in Fig. 5. The Google Multilang Speech Recognition Corpus is a subset of the Google 5M LID corpus, containing 1k manually-annotated utterances per language. We refer to [20] for more details on the training data used in our LVCSR systems.

Locale	Language	5M LID	Multilang
ar-EG	Arabic (Egypt)	✓	
ar-GULF	Arabic (Persian Gulf)	✓	
ar-LV	Arabic (Levant)	✓	
bg-BG	Bulgarian	✓	
cs-CZ	Czech	✓	
de-DE	German	✓	✓
en-GB	English (U.K.)	✓	
en-IN	English (India)	✓	
en-US	English (USA)	✓	✓
en-ZA	English (South Africa)	✓	
es-419	Spanish (Latin Am.)	✓	
es-AR	Spanish (Argentina)	✓	
es-ES	Spanish (Spain)	✓	✓
fi-FI	Finish	✓	
fr-FR	French	✓	✓
he-IL	Hebrew (Israel)	✓	
hu-HU	Hungarian	✓	
id-ID	Indonesian	✓	
it-IT	Italian	✓	✓
ja-JP	Japanese	✓	✓
ko-KR	Korean (South Korea)	✓	
ms-MY	Malay	✓	
nl-NL	Dutch	✓	
pt-BR	Portuguese (Brazilian)	✓	
pt-PT	Portuguese (Portugal)	✓	
ro-RO	Romanian	✓	
ru-RU	Russian	✓	✓
sk-SK	Slovak	✓	
sr-RS	Serbian	✓	
sv-SE	Sweden	✓	
tr-TR	Turkish	✓	
zh-CN	Chinese (Mandarin)	✓	✓
zh-TW	Chinese (Taiwan)	✓	
zh-HK	Chinese (Cantonese)	✓	

Fig. 5. List of the Google 5M LID and Google Multilang languages considered.

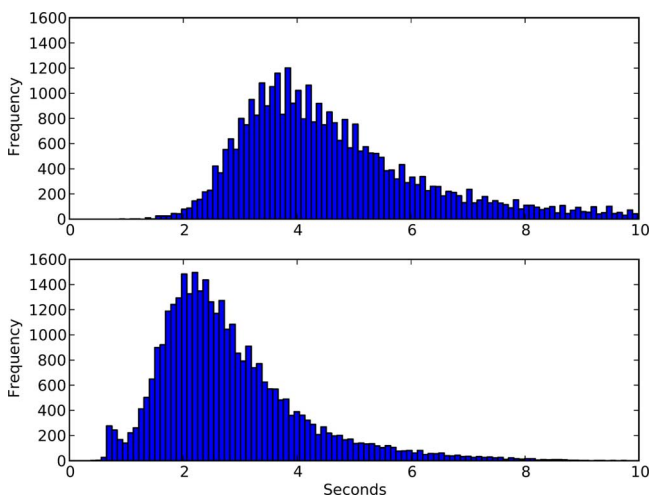

Fig. 6. Histograms of durations of the Google 5M LID and Google Multilang Speech Recognition test utterances. Original speech signals (above) and after voice activity detection (below).

It is worth mentioning that focusing on tuples with up to 8 languages also reduces the computational load on our system. In particular, running many monlingual speech recognizers in parallel is computationally expensive. In the final deployed system, we artificially restrict the number of languages that can be selected, both to improve accuracy and to protect the production system from excessive load.

3) Evaluation Metrics: Given an utterance with a known language and transcription, we evaluate the multilingual recognition system on three metrics:

- Language Id Accuracy: the ratio of the cases in which the top scored language is the true language out of a pool of N candidate languages.
- Word Error Rate (WER): the edit distance between the expected transcription and the system's transcription.
- Real-time Factor (RTF): the ratio of the speech recognition response time to the utterance duration. We examine both mean RTF (average over all utterances), and 90th percentile RTF.

Accuracy is the most direct measurement of the system's language selection performance, but it penalizes language-ambiguous utterances, consisting for example of names and loanwords, where multiple recognizers may produce similar transcripts. The WER metric typically produces a weaker penalty on utterances where the language is ambiguous. It is important to note that the language selection itself may be an important output of the system, independent of the transcription. This is true, for example, in the "multi − recognize + search + synthesize" activity pipeline shown in Fig. 1 where the selected language influences search results and voice synthesis.

For this work we developed a real-time evaluation framework that streams test utterances to the system at their natural playback rate and records responses from the system. A typical test scenario involves choosing some language tuple, sampling test utterances for each language in the tuple, and querying the system with all utterances, providing the unordered tuple as the language candidates. Thus, the system has an equal prior probability of selecting any of the languages in the tuple. The framework collects Accuracy, WER, and RTF for each utterance in the test set.

VI. Results

In this section we aim to study the perceived performance of the multi-language speech recognition system, as compared to a corresponding monolingual system in which the user selects the true spoken language for each utterance in advance. In a multi-language environment, language uncertainty will normally contribute to degraded performance on speech recognition accuracy and latency. In the following subsections we present data measuring the performance of our system along those dimensions.

A. Language Identification Performance

In previous works we showed our DNN-based LID system outperforms previous state-of-the-art approaches [12]. Over the Google 5M LID corpus, our DNN-based system surpassed previous state-of-the-art i-vector-based systems [21], [22] by 70% relative. Fig. 10 (in appendix A) depicts the confusion matrix across languages and dialects. This matrix is built by taking hard identification decisions (i.e., selecting the language with highest

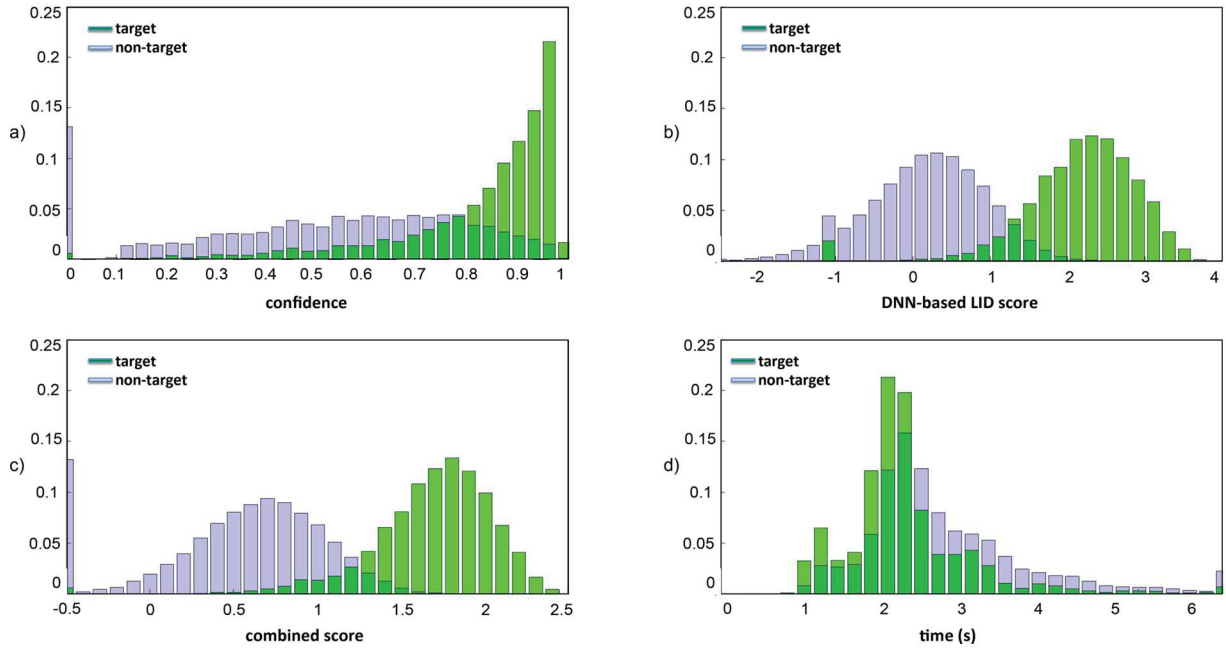

Fig. 7. Language identification over the Google Multilang Corpus. Distribution of target/non-target scores by using a) ASR confidences, b) the DNN-based LID system, c) linear weighting of confidences and DNN-based LID system; and d) as a function of the response time of the speech recognizers.

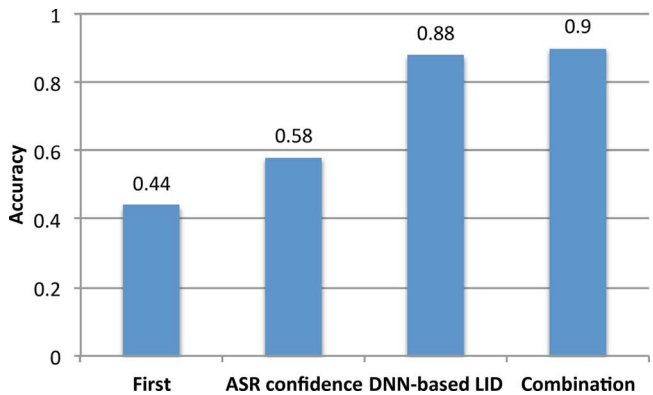

Fig. 8. Language identification performance for individual and combined systems. $N = 8$ over the Google Multilang Corpus.

associated DNN-output). On average the system achieves an accuracy of $\sim 80\%$.

However, confusion submatrices around dialects (i.e ar-EG/ar-GULF/ar-LV) illustrate the difficulty of dialect identification using only acoustic features [23]. This suggests that complementary high-level features (e.g., phonetic, phonotactic or prosodic) would be helpful, particularly when dealing with dialects or non-native accented speech. [24]–[26]. With this in mind, we use ASR confidence scores to complement the acoustic LID classifier, since these confidence scores incorporate language model cost.

Fig. 7 shows various score distributions for trials labelled either target or non-target depending on whether the score corresponds to the correct language for the given utterance. It shows the individual distributions for ASR confidence (7.a) and DNN-based LID scores after applying Z-norm [27] (7.b), as well as the linear weighted combination of the two (7.c). The set of ASR confidence scores with value zero represents trials where a speech recognizer did not emit a transcription. As expected, this occurs most often in non-target trials. The average LID accuracy achieved by these three systems (ASR confidences, DNN-based LID and Combination) for $N = 8$ over the Google Multilang Corpus is presented in Fig. 8. The DNN-only classifier outperformed ASR confidence-only classifier (88% vs 58% of accuracy), but the weighted combination achieved the best results (90% of accuracy).

In Figs. 7 and 8, we also examine ASR response times as a potential signal for language identification. Supplying a wrong-language utterance to a speech recognizer typically results in a wider effective lattice beam, reflecting a greater degree of uncertainty, and therefore a slower average response time. This effect can be observed in (7.d), where we show the distribution of final response times for target and non-target trials. Note that the non-target distribution is shifted to the right (~ 0.5 s). As a further illustration, in Fig. 8 we present language identification accuracy for a system that simply selects the first language recognizer to return a final transcription. This system achieved an accuracy of 44%, which is significantly better than chance (12.5% when $N = 8$). These results endorse the use of response times as a signal for language identification. In our system, this is exploited through the use of timeout strategies which seek to reduce latency without adversely affecting accuracy.

B. Monolingual Speech Recognition Performance

The monolingual speech recognition performance defines the ceiling for the multilingual recognizer in terms speech recognition performance and latency. We computed WER and RTF for all 8 languages in the Google Multilang Corpus, which are presented in Table I. There is significant variation in the WER metric across languages. This is due to actual variations in recognition quality, caused for example by differences in the type or amount of data used in training [20], and also synthetic variations in the WER metric computation due to text normalization and word tokenization procedures that vary

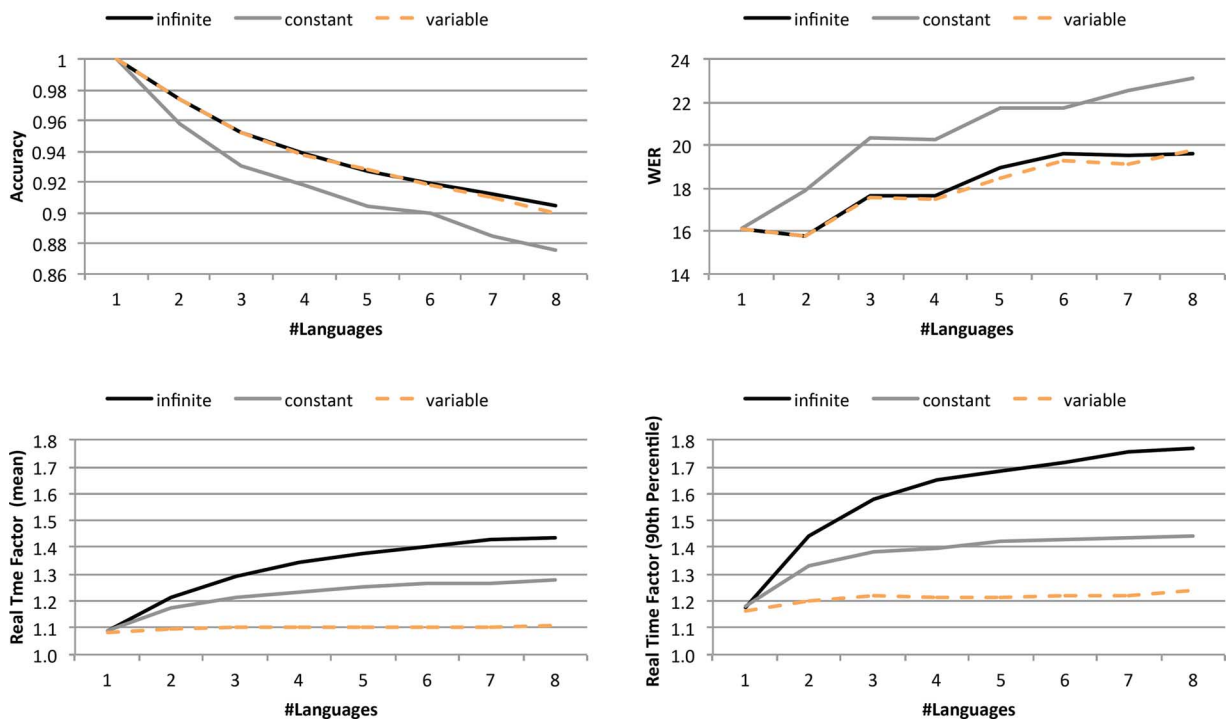

Fig. 9. Language identification accuracy, Word Error Rate (WER) and Real Time Factors (RTF) for the multilingual recognizer as a function of the number of candidate recognition languages.

by language. Thus, it is difficult to draw conclusions from a comparison of absolute WER values across languages. However, WER remains an useful metric for examining relative performance of language selection strategies over the same set of test utterances, such as the results presented in Fig. 9.

C. Multilingual Speech Recognition Performance

In order to examine the performance of the system across a range of language combinations, we adopt the following procedure using test utterances from the Google Multilang Corpus:

- Select a set of N languages.
- Sample U utterances per language.
- Select a tuple size interval $[K_{\min}, K_{\max}]$, bounded by $[1, N]$.
- Select the number of language combinations, M, to test for each tuple size in the interval.
- For each tuple size, k, randomly select M language combinations of that size, or at most the number of k-combinations (N choose k).
- For each language combination, send U utterances from each of the k languages to the multilang system, giving all languages as candidates; record the metrics described in Section V-A3.

The resulting number of trials is:

$$\sum_{k=K_{\min}}^{K_{\max}} U \times \min(M, \binom{N}{k} \times k).$$

In the results below, $N = 8$, $K = [1, 8]$, $M = 8$, $U = 1000$, for a total of 232,000 trials. We held the adjustable weighting parameters α and β constant and varied only the timeout strategy (Constant, Infinite, and Variable) as described in Section III-A.

TABLE I
WORD ERROR RATE (WER) AND REAL TIME FACTOR (RTF) OF MONOLINGUAL SPEECH RECOGNIZERS PERFORMANCE

Locale	WER	RT 50%	RT 90%
de-DE	15.58	1.08	1.17
en-US	9.38	1.09	1.19
es-ES	15.05	1.05	1.09
fr-FR	12.88	1.13	1.28
it-IT	12.19	1.07	1.14
ja-JP	20.13	1.08	1.16
ru-RU	25.72	1.11	1.21
zh-CN	18.14	1.09	1.17
Avg.	16.13	1.09	1.18

The results from all trials with same number of candidate languages were averaged to produce the graphs in Fig. 9.

Looking at the plots for the Infinite timeout strategy, we see that it provides the best Accuracy and WER out of all the strategies, but also had the worst performance on RTF. The plots for the Constant timeout strategy show improvement on RTF at the cost of Accuracy and WER.

The Variable timeout strategy was clearly the best of the three, closely tracking the Accuracy and WER curve of the Infinite timeout strategy, while performing close to the monolingual ceiling on the RTF metric. Indeed, the difference in 90th percentile RTF for $N = 1$ vs $N = 8$ was barely perceptible for the Variable timeout strategy at $\sim 6\%$ relative, compared to a very noticeable $\sim 51\%$ relative for the Infinite timeout strategy.

Under the Infinite (or Variable) strategy, recognition slowly degrades as the number of candidate languages increases. For

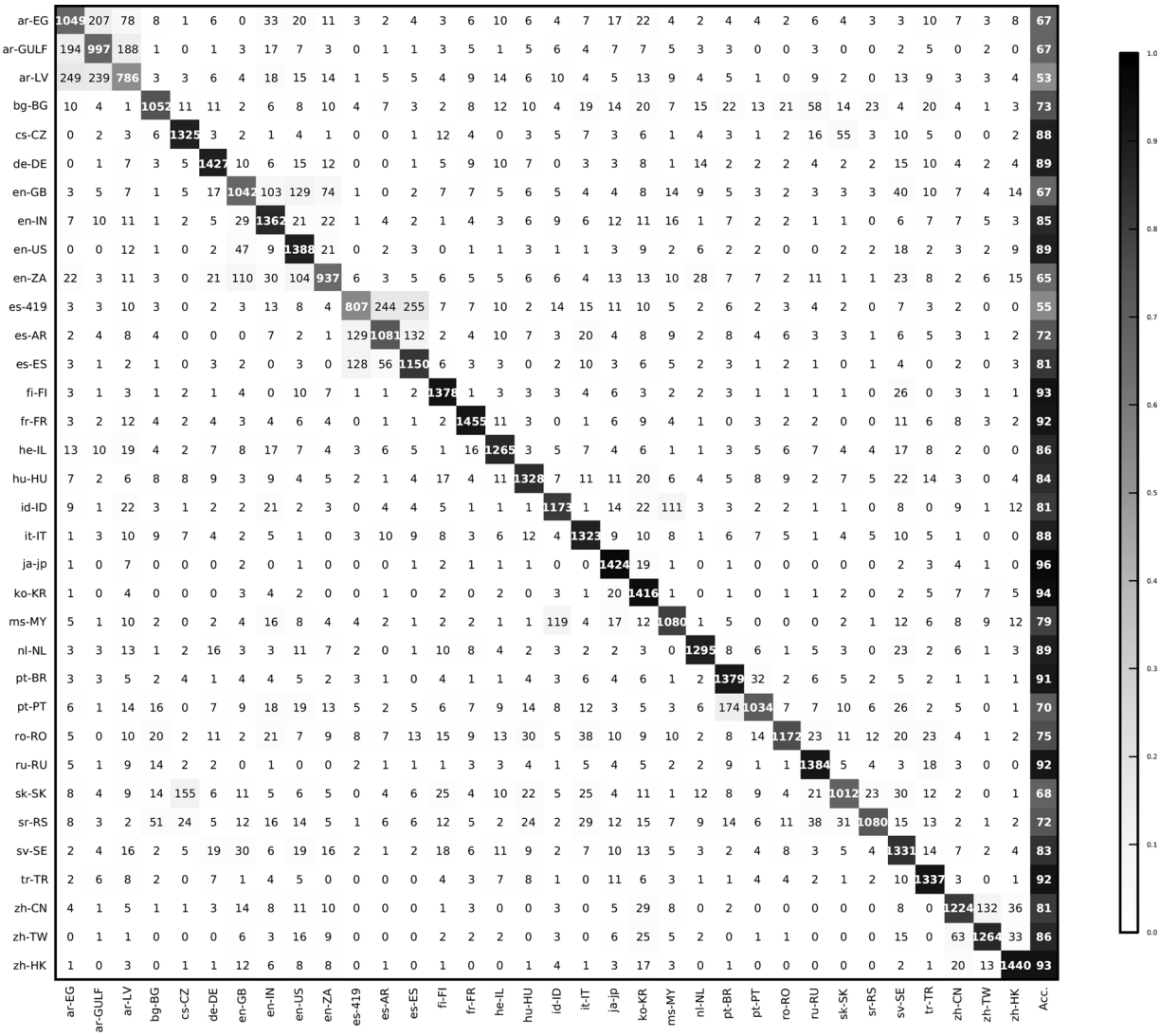

Fig. 10. Confusion matrix obtained by evaluating the DNN-based LID system on the Google 5M LID corpus.

$N = 3$ candidate languages the degradation is around $\sim 10\%$ relative $- \sim 1.5\%$ absolute-. Even for $N = 8$, the performance degradation is still acceptable, $\sim 24\%$ relative $- \sim 3.5\%$ absolute-.

VII. Summary

Through this paper we presented a novel end-to-end multilingual speech recognizer architecture developed at Google. This architecture supports multiple languages, allowing users to naturally interact with the system in several languages.

The language detection relies on the combination of a specific DNN-based LID classifier and the transcription confidences emitted by the individual speech recognizers. Thus, complementing the acoustic information exploited by the DNN-based LID classifier with the high-level information associated to the language model of the speech recognizer.

Unlike other approaches, this architecture establishes a mechanism to perform language selection in nearly real time. This allows users to transparent switch among different languages under the appearance of using a monolingual ASR.

We assessed the system in terms of both accuracy (speech recognition and LID performance) and response time in a large database including real traffic data and 34 languages. Results show that the proposed architecture is capable of managing multiple languages without significant impact on accuracy and latency compared to our monolingual speech recognizers.

References

[1] G. Tucker and A. Tucker, "A global perspective on bilingualism and bilingual education, ser. ERIC (Collection)," ERIC Clearinghouse on Languages and Linguistics, 1999 [Online]. Available: http://books.google.com/books?id=sEB5tgAACAAJ

[2] F. Grosjean, *Bilingual: Life and reality*. Harvard, MA, USA: Harvard Univ. Press, 2010 [Online]. Available: http://books.google.com/books?id=XgRum7AWOoUC

[3] D. Waggoner, "The growth of multilingualism and the need for bilingual education: What do we know so far?," *Bilingual Res. J.* vol. 17, no. 1–2, pp. 1–12, 1993 [Online]. Available: http://dx.doi.org/10.1080/15235882.1993.10162645

[4] T. Schultz and A. Waibel, "Language independent and language adaptive large vocabulary speech recognition," in *Proc. ICSLP*, 1998, vol. 1998, pp. 1819–1822.

[5] H. Lin, L. Deng, J. Droppo, D. Yu, and A. Acero, "Learning methods in multilingual speech recognition," in *Proc. NIPS*, Vancouver, BC, Canada, 2008.
[6] H.-A. Chang, Y. H. Sung, B. Strope, and F. Beaufays, "Recognizing English queries in Mandarin voice search," in *Proc. IEEE Int. Acoust., Speech, Signal Process. (ICASSP), Conf.*, May 2011, pp. 5016–5019.
[7] T. Niesler and D. Willett, "Language identification and multilingual speech recognition using discriminatively trained acoustic models," *Multilingual Speech Lang. Process.*, 2006.
[8] H. Caesar, "Integrating language identification to improve multilingual speech recognition," *Idiap, Idiap-RR Idiap-RR-24–2012*, no. 7, 2012.
[9] H. Lin, J. T. Huang, F. Beaufays, B. Strope, and Y. H. Sung, "Recognition of multilingual speech in mobile applications," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Mar. 2012, pp. 4881–4884.
[10] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems 25*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Cambridge, MA, USA: MIT Press, 2012, pp. 1232–1240.
[11] G. Heigold, V. Vanhoucke, A. W. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean, "Multilingual acoustic models using distributed deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 8619–8623.
[12] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 5337–5341.
[13] A. Mohamed, G. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 1, pp. 14–22, Jan. 2012.
[14] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
[15] D. Ciresan, U. Meier, L. Gambardella, and J. Schmidhuber, "Deep big simple neural nets excel on handwritten digit recognition," *CoRR*, vol. Abs/1003.0358, 2010.
[16] D. Yu and L. Deng, "Deep learning and its applications to signal and information processing, Exploratory DSP," *IEEE Signal Processing Mag.*, vol. 28, no. 1, pp. 145–154, Jan. 2011.
[17] Y. Bengio, *Learning deep architectures for AI*, ser. Foundations and Trends(r) in Mach. Learning. Delft, The Netherlands: Now Publishers, 2009 [Online]. Available: http://books.google.com/books?id=cq5ewg7FniMC
[18] G. Heigold, E. McDermott, V. Vanhoucke, A. Senior, and M. Bacchiani, "Asynchronous stochastic optimization for sequence training of deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Florence, Italy, 2014, pp. 5587–5591.
[19] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on cpus," in *Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop*, 2011.
[20] O. Kapralova, J. Alex, E. Weinstein, P. Moreno, and O. Siohan, "A big data approach to acoustic model training corpus selection," in *Proc. Interspeech*, 2014.
[21] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 4, pp. 788–798, Feb. 2011.
[22] D. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matejka, "Language recognition in iVectors space," in *Proc. Interspeech. ISCA*, 2011, pp. 861–864.
[23] P. A. Torres-Carrasquillo, D. E. Sturim, D. A. Reynolds, and A. McCree, "Eigen-channel compensation and discriminatively trained Gaussian mixture models for dialect and accent recognition," in *Proc. Interspeech*, 2008, pp. 723–726.
[24] F. Biadsy, "Automatic dialect and accent recognition and its application to speech recognition," Ph.D. dissertation, Columbia Univ., New York, NY, USA, 2011.
[25] W. Baker, D. Eddington, and L. Nay, "Dialect identification: The effects of region of origin and amount of experience," *Amer. Speech*, vol. 84, no. 1, pp. 48–71, 2009.
[26] G. Liu, Y. Lei, and J. H. Hansen, "Dialect identification: Impact of difference between read versus spontaneous speech," in *EUSIPCO '10*, 2003–2006.
[27] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Process.*, vol. 10, no. 1/2/3, pp. 42–54, 2000.

Javier Gonzalez-Dominguez received his M.S. degree in computer science in 2005 from Universidad Autonoma de Madrid, Spain. In 2005, he joined the Biometric Recognition Group—ATVS at Universidad Autonoma de Madrid (UAM) as a Ph.D. student. In 2007, he obtained the postgraduate master in computer science and electrical engineering from UAM and received a FPI research fellowship from the Spanish Ministerio de Educacion y Ciencia. His research interests are focused on robust speech, speaker and language recognition. He has been recipient of several awards as the Microsoft Best student paper at SIG-IL 2009 conference ant the Google best Ph.D. Thesis Award at IberSpeech 2012. He has actively participated and led several ATVS systems submitted to the NIST speaker and language evaluation recognition since 2006. During his Ph.D. pursuit he had been member of several research sites as SAIVT-QUT (2008, Brisbane, Australia), TNO (2009, Utrecht, The Netherlands) and Google Inc. Research (2010 New York, USA). Since 2012, he is an Assistant Professor at UAM. During the 2013–2014 term, he is a visiting scientist at the Speech Team in Google Research New York.

David Eustis is a Senior Staff Software Engineer in the Speech Recognition Group at Google where he develops features and infrastructure for Google's Speech Serving System. David joined Google in 2005. Prior to Speech, he worked in Google's Geographic Data Group where he led the development of Google's Geographic Entity Database, and was an early contributor to Google's Distributed Geographic Data Processing Pipelines. He received his Sc.B. degree in computer science and applied math in 2005 from Brown University and was awarded the Susan Colver Rosenberger Senior Prize in Computer Science. At Brown, he participated in research with the Brown Graphics Group at their CAVE virtual reality environment, and interned at the Los Alamos National Lab's CAVE environment as part of a BGG-LANL collaboration.

Ignacio Lopez-Moreno is a Software Engineer in the Speech Recognition Group at Google where he develops language and speaker recognition technologies and services. He received his M.S. degree in electrical engineering in 2009 from Universidad Politecnica de Madrid (UPM). He is currently pursuing his Ph.D. degree with the Biometric Recognition Group—ATVS at Universidad Autonoma de Madrid. He has participated several technology evaluations, such as NIST speaker and language recognition evaluations carried out since 2005. His research interests include speaker verification, language identification, pattern recognition, speech processing, statistics and forensic evaluation of the evidence. He has been recipient of several awards and distinctions, such as the IBM Research Best Student Paper in 2009.

Andrew Senior received his Ph.D. from Cambridge University for his thesis "Recurrent Neural Networks for Offline Cursive Handwriting Recognition." He is a research scientist at Google, New York, where he works in the speech team on deep and recurrent neural networks for acoustic modeling. Before joining Google he worked at IBM Research in the areas of handwriting, audio-visual speech, face and fingerprint recognition as well as video privacy protection and visual tracking.

Françoise Beaufays is a Senior Staff Research Engineer and Manager at Google. She received a B.S. degree in mechanical and electrical engineering from the University of Brussels (ULB) and a Ph.D. in electrical engineering from Stanford University, with a Ph.D. minor in Italian and French literature. She spent the last 20 years working on speech recognition, focusing on building products that make their users happier and more productive.

Pedro J. Moreno is a Research Scientist at Google Inc. working in the New York office. His research interests are speech and multimedia indexing and retrieval, speech and speaker recognition and applications of machine learning to multimedia. Before joining Google, he worked in the areas of text processing, image classification, bioinformatics and robust speech recognition at HP labs where he was on the lead researchers developing SpeechBot, one of the first audio search engines freely available on the web. He received a Ph.D. in electrical and computer engineering from Carnegie Mellon University and was a former Fulbright scholar.