ACCURATE CLIENT-SERVER BASED SPEECH RECOGNITION
KEEPING PERSONAL DATA ON THE CLIENT

Munir Georges1,2, Stephan Kanthak1, Dietrich Klakow2

1Nuance Communications, Automotive Speech R&D, Aachen, Germany
2Spoken Language Systems, Saarland University, Saarbrücken, Germany


ABSTRACT
In this paper, a novel technique is proposed that recognizes speech on a server but all private knowledge is processed on the client. Private knowledge could be address book entries, calendar entries or medical patient data.

The technique combines the advantage of a powerful server with almost unlimited memory and the advantage using locally available user dependent knowledge. A dynamic language model is used to recognize speech with the help of content dependent acoustic fillers on a server. The result is then recognized including user dependent knowledge on a client, e.g., a smart phone. We achieved a word error rate reduction of 17% on the Wall Street Journal Corpus.

Index Terms— Dynamic Language Model, Acoustic Filler, Client-Server Speech Recognition, Data Privacy.

1. INTRODUCTION
It would be beneficial for various speech applications to use local data such as address book entries, calendar entries or other private data. These private data are often not available on a server. This may be because of legal reasons, e.g., for medical patient data. Smaragdis et al. proposed a framework for secure speech recognition [1]. Using local data can also reduce the required server storage capacity and its software complexity for high demand speech applications. Recognition on an embedded device is often limited due to restricted computational power and memory.

We propose a novel technique that combines client and server based speech recognition through dynamic language models and acoustic fillers. There is no need to synchronize user dependent private data to achieve accurate speech recognition. All private data is recognized on the client. It enhances the recognition hypotheses from the server with suitable locally available data. This allows the use of models that are highly optimized for the use on embedded devices on the one hand. On the other hand, the server recognizer can use precise acoustic models and language models estimated on the crowd. Our novel technique can take advantage of private data that is only locally available on the client.

Fig. 1. Speech recognition hypotheses from a server are enhanced on the smart phone with user knowledge. The recognizers are connected through a Wide Area Network (WAN).

Our novel speech recognition technique uses several language models, simultaneously. Murveit et al. described a technique that uses different levels of detail between recognition passes [2]. Multiple pass search strategies were described in detail by Schwartz et al [3]. We combine statistical language models and grammars. A combining of linguistic and statistical knowledge was proposed by Moore et al [4]. Bruganara et al [5] proposed hierarchical language models. Linking several local models together according to a general one was proposed by Nasr et al [6]. The idea was further pursued using weighted transducers, e.g., by Schalkwyk et al [7] or Mohri [8]. An on-the-fly transducer nesting was proposed by Georges et al [9]. The technique described in this paper combines different language models in multiple passes on different devices. The used dynamic language model is described in Section 2. Our technique is not related to distributed language models for estimating N-grams on a grid computer using a wide Storage–Area–Network as described by Mnih et al [10] and Brants et al [11].
Each occurrence of a term from $S_k$ in $C$ is substituted with a marker $t_k \in T$ with $|T| = K$ and $T \cap W = \emptyset$. The result is a generalized corpus $C'$. We used the transducer replacement operator $R$ proposed by [13], which is described in [9] for dynamic language model estimation. Regular expressions can be used, too. The definition of $R$ is given as:

$$R : W^* \rightarrow ((W \cup T)^* \setminus S_K)^*.$$ 

A generalized $N$-gram Markov language model is then estimated on $C'$. The probability of a word sequence $w$ is given by the sequence computed by $R(w)$. The generalized language model can be formulated as:

$$P(R(w)) = \prod_{i=1}^{|R(w)|} P(R(w)_i | R(w)_{i-N+1:i-1}),$$

$$P_k(w_{m:n}) \begin{cases} 1 & m, n : R(w_{m:n}) = R(w)_i = t_k \\ \text{else.} & \end{cases}$$

$P_k$ is a conditional probability for a replaced word sequence by $R$ where the probability is 1 if no word was replaced. The model is normalized if each word sequence in $S_K$ is uniquely associated with one marker. Let $P(x|w)$ be the acoustic model. The most probable word sequence $\hat{w}$ is given by a sequence of speech features $x$ [14], [15]. Here, the fundamental formula of speech recognition becomes:

$$\hat{w} = \arg \max_{w \in W^*} P(x|w)P(R(w)).$$

The server uses the generalized language model where each $P_k$ is replaced on–the–fly with a corresponding acoustic filler as described in Section 3. The acoustic fillers are based on phoneme loop models estimated on the replaced word sequences. There are various filler alternatives described in the literature. Asadi et al. proposed fillers that where used to obtain phonetic transcriptions for modelling out of vocabulary words [16]. Jiang et al. [17] described fillers based on subword features for a vocabulary–independent word confidence measure. Fillers based on word fragments were proposed by Klakow et al. [18] and various models were described by Bazzi et al. [19]. We analysed the scope for improvement with oracle fillers as described in Section 3.

The server recognition result is used along with user grammars on the client, e.g., a smart phone to assemble an user dependent transducer. This transducer is recognized on the client as described in Section 4.

### 3. RECOGNITION ON THE SERVER

The speech recognizer on the server uses a generalized language model where user dependent word sequences were recognized with acoustic fillers. We use weighted finite state transducers, so that an on–the–fly nesting technique could be used to embed the acoustical filler models.
The generalized $N$-gram Markov language model can be represented by a weighted automaton $G_1$. The relations between phoneme sequences $P$ and words $W$ is described by a lexicon transducer $L \subseteq (P \times W)^*$. Further on, the context dependency between phonemes is given by the transducer $C$. A static search network can be assembled as follows:

$$M'_1 = \min(\det(C \circ L \circ G_1)),$$

where $\circ$ denote the composition operator [20]. $\min$, $\det$ denote the transducer operator for minimization and determination [21]. Finally, $M'_1$ is composed on-the-fly with a hidden Markov model $H$ along with the cross-word computation. This was initially described by Hori et al [22], [23] and further improved by McDonough et al [24] and Allaucen et al [25], [26]. The probability for a phoneme sequence given a sequence of speech features can be computed [27] using a token passing time synchronous Viterbi beam search [28]. Each acoustic filler is represented as a weighted transducer, sharing the same set of hidden states. There is no need to extend the acoustic model. The transducer replacing operator [27] can be used to nest the filler model into the $M'_1$ transducer. We nest the filler on-the-fly [9] each time when a marker from the generalized language model was reached.

The recognition result is an $N$-best list of sentence hypotheses [3]. The acoustic filler location is tagged and will be later used to include user dependent knowledge. Using $N$-best sentence hypotheses ensures backwards compatibility for other speech applications using the same server infrastructure. Alternatively, a lattice could be passed to the client.

### 4. RECOGNITION ON THE CLIENT

The smart phone receives recognition hypotheses from the server. The user dependent language portion is marked. This could be proper names, dates or other private data. The user data is locally available as grammar, e.g., an address book, calendar or medical recordings. These data are used for speech recognition on the client, e.g., a smart phone.

In this paper, the received $N$-best sentence hypotheses from the server were summarized in one grammar $G_2$, where each sentence ends up in one grammar rule. This is comparable to an output voting error reduction system [29] where different hypotheses from various recognizers were combined to improve the overall accuracy. Schwenk et al [30] proposed to include language model weights which may also be used for the proposed technique in this paper. Alternatively, a word lattice could be delivered by the server. Each marker in $G_2$ points to an user grammar. Similar to the recognition on the server, a transducer $M'_2$ can be assembled as:

$$M'_2 = \min(\det(C \circ L \circ G_2)).$$

The phoneme dependency model is $C$ and $L$ is the lexicon transducer. $M'_2$ is composed on-the-fly with a hidden Markov model $H$ along with the cross-word computation. A token passing time synchronous Viterbi beam search is used similar to the server recognition system. In addition, histogram pruning [14] is applied to fulfill the embedded memory requirement.

### 5. EVALUATION

The proposed system is evaluated using the Wall Street Journal Corpus [12]. We use the same decoder set-up on the server and on the client for comparable reasons. An integer value based acoustic model evaluation was used. We did not take advantage of any acoustical adaptation techniques such as MLLR etc. The SRI Language Model tool-kit [31] was used to estimate the 5k word language models with Kneser-Ney discounting [32]. We prepared the Wall Street Journal Corpus $C$ for language model training in a way that it becomes comparable to real world applications.

<table>
<thead>
<tr>
<th>Table 1. Used grammar for evaluation in Backus–Naur Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\langle \text{DAY} \rangle \ ::= \text{‘Monday’</td>
</tr>
<tr>
<td>$\langle \text{MONTH} \rangle \ ::= \text{‘January’</td>
</tr>
<tr>
<td>$\langle \text{NUM} \rangle \ ::= \langle \text{num} \rangle</td>
</tr>
<tr>
<td>$\langle \text{NUM} \rangle \ ::= \langle \text{num} \rangle</td>
</tr>
<tr>
<td>$\langle \text{MONETARY} \rangle \ ::= \langle \text{NUM} \rangle</td>
</tr>
<tr>
<td>$\langle \text{PERCENT} \rangle \ ::= \langle \text{NUM} \rangle</td>
</tr>
<tr>
<td>$\langle \text{ABBREVIATION} \rangle \ ::= \langle \text{Letter} \rangle</td>
</tr>
<tr>
<td>$\langle \text{Letter} \rangle \ ::= \langle \text{A’}</td>
</tr>
</tbody>
</table>

Imagine a short message dictation application where the local available address book, the music title collection and the calendar should be included in the recognition. Here, the user dependent knowledge $S$ is a subset of all weekdays, names of months, various number terms and abbreviations according to Table 1. We exclude all user dependent knowledge $S' \subset S$ from the corpus that occur in the set of test sentences to reduce the coverage of $S'$ from 31% down to 7%:

$$C' = \{ w \in C | \exists m, n : w_{m,n} \in S' \}.$$

This coverage seems realistic for real world applications when we analyse $N$-gram cut-off data. $C'$ is used to estimate the language models for the server only system. Each term of $S$ is replaced in the corpus $C'$ with a corresponding marker symbol from $T$. We used the grammar in Table 1 and the transducer replacement operator $R$ to build $C''$ as follows:

$$C'' = \{ w \in (W \cup T)^* | \exists w' \in C' : w = R(w') \}.$$
The generalized corpus $C''$ is used to estimate the dynamic language model. This language model is used by the server along with the acoustic fillers. Those fillers are based on phoneme loop models. Every substituted word sequence from $C''$ is used to estimate the 1-gram phoneme loop filler.

Initially, we analyzed the impact of the number of $N$-best hypotheses which were passed from the server (1st decoder) to the client (2nd decoder). We expected that this influences the recognition accuracy significantly. This has been confirmed by the experiments. Figure 3 shows the influence using 3-gram dynamic language models. We examined this behaviour also for the 2-gram and 4-gram models where we observed similar behaviour. A minimal word error rate along with tenable recognition time for the 2nd decoder was achieved using 5-best hypotheses. This experiment is denoted with an "x" in Figure 3. The pruning behavior on the 2nd decoder has a significant influence on the recognition time but nearly no influence on the accuracy for small $N$.

In this paper, phoneme loop fillers were used. Even when each model was estimated on representative data, the difference between each filler is minimal. The accuracy can be further improved using fillers which are strongly user adapted. We used oracle full word fillers estimated on the test data in the following experiment. Figure 4 illustrates the potential of improvement for dynamic 2-gram language models on representative hardware. The oracle filler outperforms the proposed phoneme loop model as expected. Further, we compared the performance with a grammar 2-gram language model where user dependent grammars were nested during decoding. This is only possible when the user data is available on the server. Our novel technique could achieve nearly the same recognition accuracy with user dependent fillers although it took a certain delay. We observed similar behaviour using the 3 and 4-gram language model set-up.

Finally, we compared our novel system with server only speech recognition. A faster recognition was achieved with 4-gram language models whereas no further accuracy improvement was observable. In summary, the recognition accuracy of the novel technique outperforms the server only system as summarized in Figure 5. No private data has to be synchronized with the server. All private data such as the address book, calendar or medical data remains on the client. The latency of the proposed technique requires a user feedback mechanism for certain applications. Here, the latency was on average 15% of the real time. The processor can stay in a battery-saving mode for the most time. A word error rate reduction of 17% was achieved for the 3-gram dynamic language model set-up.

**6. SUMMARY**

We propose a technique for speech recognition on a client and server where no private data has to be available on the server. Private data could be an address book, a private calendar or some medical patient data. A dynamic language model is used on the server along with acoustic fillers. The recognition result is then combined with user dependent knowledge on the client, e.g., on a smart phone.

We have shown that the proposed technique can improve speech recognition on the Wall Street Journal corpus. An average latency of 15% was observed compared to real time recognition and, in the same time, a word error rate reduction of 17% was achieved.
7. REFERENCES


