# Training RNN Language Models on Uncertain ASR Hypotheses in Limited Data Scenarios

Imran Ahamad Sheikh, Emmanuel Vincent, Irina Illina

# Training RNN Language Models on Uncertain ASR Hypotheses in Limited Data Scenarios

Imran Sheikh, Emmanuel Vincent, Irina Illina

*Université de Lorraine, CNRS, Inria, Loria*
*F-54000 Nancy, France*

**Abstract**

Training domain-specific automatic speech recognition (ASR) systems requires a suitable amount of data comprising the target domain. In several scenarios, such as early development stages, privacy-critical applications, or under-resourced languages, only a limited amount of in-domain speech data and an even smaller amount of manual text transcriptions, if any, are available. This motivates the study of ASR language models (LMs) learned from a limited amount of in-domain speech data. Early works have attempted training of n-gram LMs from ASR N-best lists and lattices but training and adaptation of recurrent neural network (RNN) LMs from ASR transcripts has not received attention. In this work, we study training and adaptation of RNN LMs using alternate and uncertain ASR hypotheses embedded in ASR confusion networks obtained from target domain speech data. We explore different methods for training the RNN LMs to deal with the uncertain input sequences. The first method extends the cross-entropy objective into a Kullback–Leibler (KL) divergence based training loss, the second method formulates a training loss based on a hidden Markov model (HMM), and the third method performs training on paths sampled from the confusion networks. These methods are applied to limited data setups including telephone and meeting conversation datasets. Performance

*Email for correspondence: mranahmd@gmail.com
*Email address:*
`imran.sheikh@inria.fr,emmanuel.vincent@inria.fr,irina.illina@loria.fr` (Imran Sheikh, Emmanuel Vincent, Irina Illina)

is evaluated under two settings wherein no manual transcriptions or a small amount of manual transcriptions are available to aid the training. Moreover, a model adaptation setting is also evaluated wherein the RNN LM is pre-trained on an out-of-domain conversational corpus. Overall the sampling method for training RNN LMs on ASR confusion networks performs the best, and results in up to 12% relative reduction in perplexity on the meeting dataset as compared to training on ASR 1-best hypotheses, without any manual transcriptions.

*Keywords:* automatic speech recognition; language models; recurrent neural networks; confusion networks

---

## 1. Introduction

Automatic speech recognition (ASR) is now available as easy-to-integrate commercial APIs (Kim et al., 2019) as well as open-source platforms (Rizk, 2019). A typical commercial ASR solution is powered by an acoustic model (AM) and a language model (LM), which are trained on large amounts of speech data collected from end users and on large amounts of text data including the corresponding manual text transcriptions. Open-source ASR alternatives are backed by AMs and LMs trained on publicly contributed corpora of read speech or monologues instead (Ardila et al., 2020; Pratap et al., 2020). While AMs are portable across different application domains, such as travel, shopping, medical, etc., LMs that do not match the target domain result in a poor ASR performance. Hence, training or adaptation of LMs on in-domain data remains essential for application-specific ASR systems (Bellegarda, 2004).

Existing ASR LM adaptation approaches can be grouped into three overlapping categories; viz. (a) combination of out-of-domain and in-domain texts or LMs (Pusateri et al., 2019; Huang et al., 2020), (b) adaptation of LMs to in-domain text or features (Tam and Schultz, 2009; Deena et al., 2016; Gangireddy et al., 2016; Li et al., 2018), and (c) combination and optimization of multi-domain LMs to the target domain (Ballinger et al., 2010; Irie et al., 2018; Raju et al., 2018). Some of these approaches rely on offline training (Pusateri et al., 2019; Huang et al., 2020; Deena et al., 2016; Irie et al., 2018), while others perform a dynamic adaptation during test (Tam and Schultz, 2009; Ballinger et al., 2010; Gangireddy et al., 2016; Li et al., 2018). The offline training approaches have been tried with training text comprising manually verified transcriptions of hundreds of hours of speech from the target domain and sometimes additional text from other domains. Such manually verified text resources are scarce or even unavailable for most languages or applications. Moreover, in several use cases, the amount of in-domain speech data may be limited too. This is the case for instance in the early development stages of a new application where data is elicited from developers or beta-testers, for privacy-critical applications where no data is collected from the end users, or for under-resourced languages where the amount of data collected from the end users increases slowly over time. Exploiting such limited data is therefore crucial. This motivates us to study training and adaptation of LMs from a limited amount (25 - 50 hours) of in-domain speech data.

Early works have demonstrated that ASR transcriptions of spoken utterances can be successfully used for adaptation of traditional n-gram LMs (Niesler and Willett, 2002; Bacchiani and Roark, 2003; Tur and Stolcke, 2007). Going beyond the 1-best ASR transcripts, prior works have used web search and retrieval methods to augment LM training data (Langzhou Chen et al., 2003; Meng et al., 2010; Lecorvé et al., 2012), filtering of ASR transcripts based on confidence scores (Haznedaroglu and Arslan, 2014; Xie and Chen, 2013) and, more interestingly, training from ASR N-best lists and lattices (Bacchiani et al., 2006; Kuznetsov et al., 2016; Levit et al., 2018). Apart from these works on n-gram LMs, discriminative LMs have relied on ASR decoded hypotheses (Xu et al., 2009; Çelebi et al., 2012). However, training and adaptation of modern recurrent neural network (RNN) LMs from ASR transcripts has not received attention. The limited prior works along this direction have studied test time adaptation (Gangireddy et al., 2016; Li et al., 2018) or contextualisation (Deena et al., 2016) of RNN LMs based on 1-best ASR transcripts.

We explore training and adaptation of RNN LMs using alternate and uncertain ASR hypotheses obtained from target domain speech data. ASR outputs in the form of lattices and confusion networks (aka sausages) (Xu et al., 2011) carry information on competing ASR hypotheses, and often contain alternate hypotheses which have lower error rates compared to the 1-best ASR transcript. Early works have shown their effectiveness in intent classification (Hakkani-Tür et al., 2006; Yang and Liu, 2015) and machine translation (Zhang and Kikui, 2006; Matusov et al., 2005) tasks. More recent works on these tasks have extended RNNs to ASR lattices (Ladhak et al., 2016; Sperber et al., 2017; Huang and Chen, 2020) and confusion networks (Jagfeld and Vu, 2017; Pal et al., 2020). These works follow an encoder-decoder architecture, in which an encoder RNN first encodes the ASR lattice or confusion network into a vector representation. The encoded representation is then fed to the decoder to classify the intent or to generate the translated text. In contrast to these tasks, training RNN LMs on ASR decoded lattices or confusion networks of unlabeled speech does not have a completely certain, or manually verified, target to guide the training.

In this work, we propose three different methods to learn RNN LMs from ASR confusion networks, with the motivation to exploit the uncertainties captured in ASR confusion networks. The first method extends the cross-entropy objective into a Kullback–Leibler (KL) divergence based training loss function, the second method formulates a training loss based on a hidden Markov model (HMM), and the third method performs cross-entropy based training on paths sampled from the confusion networks.[1] We apply these methods to limited data setups including telephone and meeting conversation datasets. Performance is evaluated in two settings wherein no manual transcriptions or a small amount of manual transcriptions are available to aid the training. Moreover, we also evaluate these methods in a model adaptation setting wherein the RNN LM is pre-trained on an out-of-domain conversational corpus.

The rest of the paper is organised as follows. Section 2 starts with a formu-

[1] While we present and evaluate these methods on ASR confusion networks, the methods can be extended to ASR lattices.

lation of standard RNN LMs followed by an introduction and description of the three proposed methods for training RNN LMs using ASR confusion networks. The experimental setup used to evaluate the proposed methods is described in Section 3. Section 4 presents a detailed discussion on the performance of the different RNN LM training methods under different settings. This is followed by a conclusion in Section 5.

## 2. Learning RNN LMs from uncertain word sequences

RNN LMs have led to state-of-the-art lexicon based ASR systems (Mikolov et al., 2010; Sundermeyer et al., 2015). Likewise, they help to achieve the best performance with state-of-the-art lexicon free end-to-end ASR systems (Toshniwal et al., 2018). In particular, RNN LMs with long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) or gated recurrent unit (GRU) (Cho et al., 2014) layers are most commonly used in ASR. In the following, we first present training of RNN LMs on usual text transcriptions and then describe the proposed methods to train RNN LMs on ASR confusion networks. For the sake of legibility, we use notations similar to those for classical RNN LMs. The underlying operations can easily be extended to LSTM- and GRU-based LMs.

### *2.1. Training on usual text transcriptions*

Given a text corpus containing word sequences $W = (w_1, w_2, ...w_t, ..., w_N)$, the goal of LM training is to learn a model distribution $Q(.)$ that is as close as possible to the empirical distribution $P(.)$ of the corpus. This can be achieved by minimizing the cross-entropy

$$\mathbb{H}(P, Q) = -\sum_{W} P(W) \log Q(W). \tag{1}$$

An RNN LM consisting of $L$ recurrent layers with weight matrices $\{\theta^l_{\text{in}}, \theta^l_{\text{hid}}, \theta_{\text{out}}\}$ works as follows[2]:

$$h^l_t = \sigma(\theta^l_{\text{hid}}\ h^l_{t-1} + \theta^l_{\text{in}}\ x^l_t) \tag{2}$$

$$q(w_{t+1}|h^L_t) = \text{softmax}(\theta_{\text{out}}\ h^L_t) \tag{3}$$

where $x^1_t$ is the word embedding vector of the $t$-th word $w_t$, $h^l_t$ is the $l$-th layer hidden state vector which encodes the history or context until $t$ and $x^l_t = h^{l-1}_t$ for $l > 1$, and $\sigma$ is the non-linear function applied at every layer in the RNN. The softmax function estimates the vector of history dependent word-level LM probabilities $q(w_{t+1}|h^L_t)$. During RNN LM training, the objective is to learn the set of parameters $\Theta = \{\theta^l_{\text{in}}, \theta^l_{\text{hid}}, \theta_{\text{out}}\}$ that minimizes the cross-entropy loss. If the $(t+1)$-th word in the observed training sequence is $w_{t+1} = v^j$, where $v^j$ denotes the $j$-th word in the LM vocabulary, then the empirical distribution $P(.)$ satisfies $p(w_{t+1} = v^j|w_1, w_2, ..., w_t) = 1$ and $p(w_{t+1} = v^k|w_1, w_2, ..., w_t) = 0\ \forall k \neq j$. Hence, expressing both $P(.)$ and $Q(.)$ via the chain rule, the sum of sequence-level costs in (1) simplifies into a sum of word-level costs:

$$\widehat{\Theta} = \arg\min_{\Theta} \sum_t - \log q(w_{t+1} = v^j|h^L_t) \tag{4}$$

where $q(w_{t+1} = v^j|h^L_t)$ is the $j$-th element of the vector $q(w_{t+1}|h^L_t)$.

### *2.2. Training on ASR confusion networks*

Figure 1 shows the graphical representation of an ASR confusion network. The confusion network consists of a sequence of confusion bins, where each bin contains one or more arcs that represent alternative word hypotheses. Each arc in a bin has an associated posterior probability or score, implying that some word hypotheses are more likely than others.

A typical RNN LM, as defined above, cannot be trained on uncertain word sequences since it assumes a single word input $x^1_t$ at each step $t$ in (2) and a single word output $v_j$ at step $t + 1$ in (4). Prior works have trained RNNs on

[2] The bias vectors of the RNN are excluded for the sake of legibility.

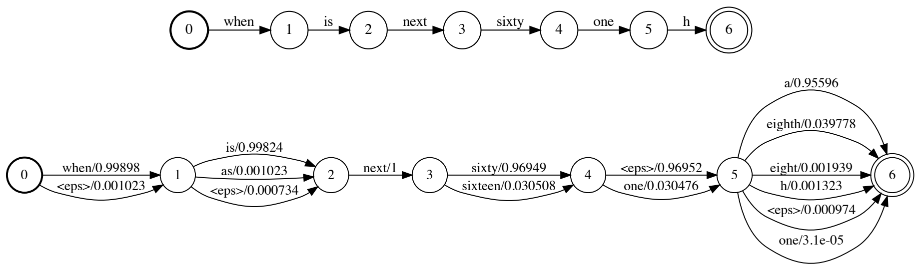


Figure 1: Graphical representation of an ASR confusion network (bottom) and the corresponding reference transcription (top).

ASR confusion networks for classification tasks (Jagfeld and Vu, 2017; Pal et al., 2020). Following these works, we can adopt the solution to compute one hidden state vector $h^1_{t,i}$ corresponding to each arc $i$ in each confusion bin and then to pool over the hidden state vectors as follows:

$$h^1_{t,i} = \sigma(\theta^1_{\text{hid}}\ h^1_{t-1} + \theta^1_{\text{in}}\ x^1_{t,i}) \tag{5}$$

$$h^1_t = \text{pool}_i(h^1_{t,i}) \tag{6}$$

where average, weighted-sum or even attention based pooling can be used. The hidden states of the following layers $l > 1$, if any, and the output probabilities are then computed as in (2) and (3).

However, the RNN LM training loss function must also be updated to handle the multiple output arcs $w_{t+1,j}$ possible at the next step $t+1$. This is the main problem with training RNN LM on confusion networks, or other decoder graphs with alternative hypotheses, and it remains unaddressed in the literature. We propose three different methods to address the RNN LM training objective. It must be noted that our methods only modify the training process and the loss function. Interestingly, the forward propagation through the RNN as well as the loss functions of each of these methods simplify back to those of a standard RNN LM when each confusion bin involves involves a single word hypothesis. Thus, computation of the LM probabilities on a simple word sequence remains identical to the standard RNN LM.

*2.2.1. From cross-entropy to KL divergence*

The cross-entropy between the LM distribution $Q$ and the empirical distribution $P$ of the corpus, formulated in (1), can be re-written as

$$\mathbb{H}(P, Q) = \mathbb{H}(P) + D_{\mathrm{KL}}(P||Q) \tag{7}$$

where $\mathbb{H}(P)$ is the entropy of $P$ and $D_{\mathrm{KL}}(P||Q)$ is the KL divergence of $Q$ from $P$. This leads us to the plausibility of using a KL divergence based loss function for training RNN LMs on ASR confusion networks. A training objective which aims to minimize the KL divergence between the RNN LM predictions $q(w_{t+1} = v^j|h_t^L)$ and the confusion bin posteriors $p(w_{t+1} = v^j|S)$ can be formulated as

$$\widehat{\Theta} = \arg\min_{\Theta} \sum_t D_{\mathrm{KL}}\left(p(w_{t+1}|S) \;||\; q(w_{t+1}|h_t^L)\right) \tag{8}$$

$$= \arg\min_{\Theta} \sum_t \sum_{v^j \in V} p(w_{t+1} = v^j|S) \log \frac{p(w_{t+1} = v^j|S)}{q(w_{t+1} = v^j|h_t^L)} \tag{9}$$

where $V$ denotes the RNN LM vocabulary and $S$ denotes the observed speech signal which led to the confusion bin posteriors. Interestingly, Huang and Chen (2020) have shown that pre-training a bidirectional LSTM-RNN classifier with a KL divergence loss function can improve the performance on intent and dialog act classification tasks. This further motivates us to evaluate the effectiveness of RNN LMs trained on ASR confusion networks using the KL divergence loss.

*2.2.2. A hidden Markov model formulation*

The KL divergence based training method discussed above tries to bring the model predictions close to the posteriors associated with the alternative ASR hypotheses, during each training iteration. In contrast, a probabilistic training method can explicitly take into account the uncertainty in the training sequences along with their degree of uncertainty. Aiming for such a method, we draw inspiration from the hidden Markov model (HMM) (Rabiner and Juang, 1986) which can model the probability distribution of a hidden sequence given a sequence of noisy or uncertain observations (Gales and Young, 2007; Ozerov et al., 2013).

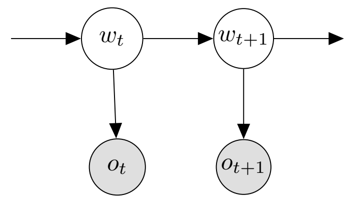


Figure 2: First order hidden Markov model.

When learning LMs from ASR confusion networks, one can imagine that the confusion bins are the sequence of observations and the HMM state transition probabilities correspond to the LM probabilities. For instance, the first order HMM, as depicted in Figure 2, would be equivalent to a bi-gram LM. The total probability of the observations $O = o_1, o_2, ..., o_t, ..., o_T$ can be obtained by summing over all possible hidden state sequences as

$$p(O|\Theta) = \sum_W p(O|W)\, p(W) \tag{10}$$

$$= \sum_{v^i, v^j \in V} \prod_t p(w_{t+1} = v^j | w_t = v^i)\, p(o_{t+1}|w_{t+1} = v^j) \tag{11}$$

where $V$ denotes the bi-gram LM vocabulary and $1 \leq i, j \leq |V|$. The total probability can be efficiently computed using the forward algorithm (Rabiner and Juang, 1986) as

$$\alpha_{t+1}(v^j) = \sum_{v^i \in V} \alpha_t(v^i)\, p(w_{t+1} = v^j | w_t = v^i)\, p(o_{t+1}|w_{t+1} = v^j) \tag{12}$$

$$p(O|\Theta) = \sum_{v^j \in V} \alpha_T(v^j). \tag{13}$$

In the case when the training data itself gives an indication of the hidden state via the posterior probability $p(w_{t+1} = v^j | o_{t+1})$, we can apply Bayes rule and express the state observation likelihood as

$$p(o_{t+1}|w_{t+1} = v^j) = p(w_{t+1} = v^j | o_{t+1}) \frac{p(o_{t+1})}{p(w_{t+1} = v^j)}. \tag{14}$$

The evidence $p(o_{t+1})$ can be treated as constant and the prior $p(w_{t+1} = v^j)$ can be estimated by averaging $p(w_{t+1} = v^j | o_{t+1})$ over entire training dataset.

Accordingly, the HMM forward probability equation (12) can be rewritten (up to a multiplicative constant) as

$$\alpha_{t+1}(v^j) = \sum_{v^i \in V} \alpha_t(v^i)\, p(w_{t+1} = v^j | w_t = v^i)\, \frac{p(w_{t+1} = v^j | o_{t+1})}{p(w_{t+1} = v^j)}. \tag{15}$$

The above HMM forward probability equation (15) can be extended to RNN LMs on ASR confusion networks. The RNN LMs compute next word probabilities $q(w_{t+1} = v^j | h_t^L)$ based on the long history or context in the hidden states of the RNN, as in (3), as opposed to the bi-gram transitions in HMM. Thus, the forward probability computation in an HMM based RNN LM can be expressed as[3]

$$\alpha_{t+1}(v^j) = \sum_{v^i \in V} \alpha_t(v^i)\, q(w_{t+1} = v^j | h_t^L)\, \frac{p(w_{t+1} = v^j | S)}{p(w_{t+1} = v^j)} \tag{16}$$

where $p(w_{t+1} = v^j)$ is obtained by averaging $p(w_{t+1} = v^j | S)$ over the entire training dataset. It should be noted that the above forward probability equation (16) scales the model predictions $q(w_{t+1} = v^j | h_t^L)$ by the posterior probabilities in the confusion bin $p(w_{t+1} = v^j | S)$, thus taking into account the uncertainty in the training sequences. Finally, the forward probability can be used to formulate the RNN LM training objective as

$$\widehat{\Theta} = \arg\min_{\Theta} - \log \sum_{v^j \in V} \alpha_T(v^j). \tag{17}$$

*2.2.3. Sampling based approach*

The RNN LMs based on KL divergence loss and HMM formulation, discussed in the previous sections, account for all the competing hypotheses from an ASR confusion network in each forward-backward propagation of the RNN. Another alternative to account for the competing hypotheses is to sample one path at a time from the ASR confusion network for each forward-backward propagation.

[3] In the special case when the RNN LM has a single layer ($L = 1$), the term $h_t^L$ can be replaced by $h_{t,i}^1$ and the approximation induced by the pooling of hidden states in (6) can be avoided using the forward algorithm. However, in our experiments we found that pooling resulted in a better performance.

To sample a complete path $\bar{W}$, one arc $\bar{w}_t$ can be sampled at a time based on the posterior probabilities of the arcs in each confusion bin as

$$\bar{w}_t \sim p(w_t|o_t). \tag{18}$$

It must be noted that sampling based on the posterior probabilities implicitly accounts for the uncertainty in the ASR hypotheses. Given a sampled path from the confusion network, the RNN LM can be trained with the standard cross entropy objective in (4). Each training epoch sees one possible path from the ASR confusion network of each utterance. The random path for each utterance is redrawn at each epoch.

The sampling based approach can be seen as a data augmentation approach for training the RNN LM. In Section 2.3.1, we recall a work on data noising in RNN LMs and show a correspondence between our sampling based approach for training from ASR confusion networks and data noising. Furthermore, the data noising scheme is evaluated along with our sampling method in the experiments.

### *2.3. Data noising and smoothing in RNN LMs*

One of the problems with learning LMs from limited amount of training data is the ability to handle rare and unseen sequences. Traditional n-gram LMs cope with this problem through discounting and smoothing techniques, the most popular one in ASR LMs being modified interpolated Kneser-Ney smoothing (Chen and Goodman, 1996). RNN LMs partly address this problem through distributed word representations and without explicitly dealing with word counts. However, overfitting due to data sparsity remains and hence RNN LMs use standard neural network regularisation methods like dropout (Srivastava et al., 2014) to alleviate this problem. In contrast, data noising and implicit augmentation methods can be more effective for RNN LMs.

#### *2.3.1. Bigram Kneser-Ney noising (KNN) in RNN LMs*

Xie et al. (2017) have presented a theoretical correspondence between data noising and smoothing. They showed that data noising motivated by bigram

Kneser-Ney smoothing results in more effective RNN LMs, as compared to blank noising (word dropout), unigram noising (word replacement) and other regularisation methods. Bigram Kneser-Ney noising (KNN) in RNN LMs applies noise to an input-output token pair $x = w_t, y = w_{t+1}$ with some noising probability $\gamma$. The token pair is replaced with the noised versions $\bar{x}, \bar{y}$ as

$$\begin{aligned} \bar{x} &\sim \text{Categorical}(\rho) \\ \bar{y} &\sim \text{Categorical}(\rho) \end{aligned} \tag{19}$$

where $\rho$ is the proposal distribution. Denoting $N_{1+}(v^j, \bullet)$ and $N_{1+}(\bullet, v^j)$ as the number of distinct bigrams beginning and ending with a word $v^j$ from the vocabulary (Chen and Goodman, 1996), respectively, the noising probability $\gamma$ and the proposal distribution $\rho$ are obtained as

$$\gamma \leftarrow \gamma_0 \frac{N_{1+}(v^j, \bullet)}{c(v^j)} \tag{20}$$

$$\rho \propto N_{1+}(\bullet, v^j) \tag{21}$$

where $c(v^j)$ denotes the total count of vocabulary word $v^j$ in the corpus, and $0 \leq \gamma_0 \leq 1$ is the noising hyper-parameter which is chosen empirically based on performance on the held out development set. The choice of $\gamma$ is motivated by absolute discounting and encourages noising of unigrams that precede many possible other tokens. At the same time it discourages noising of common unigrams. $\rho$ proposes unigrams that complete a large number of bigrams. It should also be noted that the input-output token pair is noised at each step $t$.

*2.3.2. Sampling from ASR confusion networks as data noising*

Our sampling based approach to train RNN LMs from ASR confusion networks, as discussed in Section 2.2.3, samples word alternatives from the competing arcs in each ASR confusion bin. Apart from the bigram KN noising scheme, Xie et al. (2017) discussed the unigram noising scheme which samples word alternatives based on unigram statistics. We see a connection between our sampling approach and their unigram noising scheme. We can state that the data noising criterion in (19) remains unchanged but the noising probability and proposal distribution for our sampling approach translate into

$$\gamma = 1 \tag{22}$$

$$\rho = p(w_t|o_t) \tag{23}$$

following (18). In other words, the posteriors in a confusion network bin form a time-varying proposal distribution for noising.

## 3. Experimental setup

We evaluate the proposed methods for training RNN LMs on ASR confusion networks on two conversational speech datasets extracted from the AMI and Verbmobil corpora. As detailed in the following, the conversations from these two datasets significantly differ in their characteristics. This enables us to present a more thorough evaluation of the training methods.

### *3.1. Datasets*

To simulate realistic limited data scenarios, the datasets are split into four disjoint subsets: a bigger split representing speech without manual transcriptions, a relatively smaller split containing speech with manual transcriptions, a development set and a test set. Table 1 presents the two datasets and their splits used in our evaluation setup. A brief description of these datasets is presented in the following.

#### *3.1.1. Verbmobil English conversations*

The Verbmobil corpus (Burger et al., 2000) includes about 25 hours of English conversations wherein the two participants negotiate and agree upon an appointment schedule and/or travel plan. We have split the entire Verbmobil English speech corpus into four splits, ensuring that there are no overlapping speakers or conversations across the four splits. We consider ∼5 hours of the Verbmobil corpus as a labeled training set and ∼19 hours as an unlabeled training set. The development and test sets consist of ∼2 and ∼4 hours of speech, respectively. The average length of a turn in these dialogues is 20 words.

Table 1: Datasets and splits.

| Split | Verbmobil English conversation (VM) dataset | | AMI scenario-only meeting dataset | |
|---|---|---|---|---|
| | hours | tokens | hours | tokens |
| Training manually labeled | 5.23 | 18 k | 9.48 | 90 k |
| Training unlabeled | 19.36 | 80 k | 37.24 | 387 k |
| Development | 2.14 | 7.5 k | 9.77 | 100 k |
| Test | 3.88 | 15 k | 10.34 | 105 k |

#### *3.1.2. AMI scenario-only meetings*

The AMI meeting corpus (Renals et al., 2007) has a *scenario-only* meeting subset in which the participants play different roles in a design project. We use the original scenario-only meeting subset[4] and split its training part *SA* into two sub-parts representing the labeled and unlabeled training sets. Specifically, meetings ES2010, ES2016, IS1005, IS1007, TS3010, TS3011 of SA form our labeled training set and the remaining meetings of *SA* form our unlabeled training set. The development part *SB* and evaluation part *SC* of the original scenario-only meetings are used as our development and test sets, respectively. The average length of a turn in these meeting conversations is 8 words.

### *3.2. Evaluation settings*

The methods for training RNN LMs on ASR confusion networks are evaluated in four different settings. These settings represent practical language model training scenarios wherein:

- in-domain speech data is unlabeled and limited, and/or
- a small amount of manually transcribed in-domain speech is available, and/or

[4] `https://groups.inf.ed.ac.uk/ami/corpus/datasets.shtml`

- a good amount of out-of-domain text transcriptions is available.

#### *3.2.1. Without and with manual labeled training data*

In the first evaluation setting, the RNN LM is trained only on the ASR 1-best hypotheses or the ASR confusion networks obtained from the unlabeled in-domain speech data available for training. This setting represents the practical situation wherein no manual transcriptions of in-domain speech are available for training. The second setting represents the situation wherein a small amount of manually labeled in-domain speech is available, and the RNN LM is trained on both manually labeled in-domain speech and the ASR 1-best hypotheses or confusion networks obtained from the unlabeled in-domain speech. Adding manually labeled in-domain speech is expected to result in better performance, not only because of the manual labels but also because this increases the total amount of training data. Rather than the absolute performance, we are interested in the effect of including some in-domain labeled data on the proposed training methods.

#### *3.2.2. Training vs. adaptation*

When the in-domain training data is limited, RNN LMs can be pre-trained on a larger amount of out-of-domain text data and then adapted to the available in-domain text (Ma et al., 2017). Our evaluation of the different methods of training RNN LMs on ASR confusion networks is also extended to an adaptation setting. We use the transcriptions of the Switchboard (Godfrey et al., 1992) English corpus[5] as the out-of-domain spoken conversational text. We adopt a simple but competitive domain adaptation method wherein the RNN LM is first trained on the combination of the out-of-domain and in-domain datasets, namely Switchboard and Verbmobil (SWB+VM) or Switchboard and AMI (SWB+AMI), and the entire RNN LM is then fine-tuned on the respective in-domain dataset.

[5] fetched from `http://www.isip.piconepress.com/projects/switchboard/releases/switchboard_word_alignments.tar.gz`

### *3.3. RNN LM configuration and training details*

#### *3.3.1. LM vocabulary*

In our experiments, the RNN LMs trained on the combined out-of-domain and in-domain datasets have a vocabulary of 12,119 and 12,918 for SWB+VM and SWB+AMI, respectively. RNN LMs trained only on AMI as well as RNN LMs adapted to AMI retain the SWB+AMI vocabulary. Unlike AMI, the VM corpus vocabulary does not span over the entire SWB vocabulary. Hence, RNN LMs trained only on VM as well as those adapted to VM retain a VM specific vocabulary. Perplexities reported on the VM development and test sets in the adaptation setting are computed after reducing the RNN LM embedding matrix accordingly.

#### *3.3.2. Model settings and hyper-parameters*

All experiments use LSTM-RNN LMs, i.e. RNN LMs with LSTM cells, as they have been shown to outperform the original RNN LMs and to be more effective than GRU-RNN LMs (Irie, 2020). All RNN LMs in our experiment have a single RNN layer, as training only on limited in-domain text highly overfits with more than one RNN layer. The input embedding matrix and the output word embedding matrix in our LSTM-RNN LM are tied and shared, as this results in a significant reduction of model parameters and improves the LM perplexity (Press and Wolf, 2017; Inan et al., 2017). During training, each utterance is treated independently without sharing hidden states or context across utterances.

The dimension of the word embeddings and the weight matrices in the LSTM cells is set to 64 for models trained only on in-domain text and 128 for models adapted to in-domain text after pre-training with SWB data. Increasing the dimensionality beyond 64 did not give significant perplexity improvements with the small amount of in-domain training text. Models are trained using the Adam optimizer and training is controlled using an early stopping criterion which monitors perplexity on the development set. The noising probability hyper-parameter $\gamma_0$ is chosen among 0.25, 0.5, 0.75 based on the perplexity measured on the development set.

As discussed in Section 2.2, the hidden states of RNN LMs applied on ASR confusion networks can be obtained by applying pooling over the RNN hidden states corresponding to arcs from the confusion bin at the previous step. We experimented with average, weighted-sum, max and 1-best pooling, wherein the hidden state corresponding to the arc with the highest score is chosen. We found that 1-best pooling resulted in the best performance for both KL and HMM based training.

#### *3.3.3. ASR setup*

Our ASR system is based on the Kaldi Chain acoustic model architecture with Time Delay Neural Network (TDNN) layers and i-vectors for speaker adaptation (Povey et al., 2016). In case of the VM dataset, the AM and the seed LM are trained on the labeled training set. The AM has a TDNN architecture with splices {-2,-1,0,1,2} {-1,0,1} {-1,0,-1} {-3,0,3} {-3,0,3} {-6,-3,0} at each successive layer with 512 dimensions. The inputs are 40 Mel-frequency cepstral coefficients and 100 dimensional online i-vectors. The i-vector extractor is trained on the combined labeled and unlabeled datasets. The seed LM is a standard 3-gram LM with interpolated Kneser-Ney smoothing. The VM seed AM and LM result in a Word Error Rate (WER) of 39.52% and 39.77% on the VM development and test sets, respectively.

For experiments on the AMI dataset we use the ASpIRE[6] chain model, trained on the Fisher English corpus, and the accompanying pre-compiled decoding graph. The motivation behind this choice was to evaluate the performance with a strong pre-trained AM and LM, in contrast to the VM setup. The ASpIRE chain models result in 33.15% and 35.82% WER on the AMI development and test sets, respectively.

[6] `http://kaldi-asr.org/models/m1`

## 4. Results and discussion

### 4.1. Perplexity evaluation

The perplexities achieved by the different LSTM-RNN LMs on the VM and AMI scenario meeting datasets are presented in Table 2 and Table 3, respectively. Each of these tables is horizontally divided into two halves representing evaluation without vs. with manually labeled training data. The column groupings represent evaluation of the pre-trained LM, adapted LM and LM trained only on the in-domain data. Perplexities are reported on the development and test splits of the respective datasets. Following the recommendations by Dror et al. (2018) we use the Wilcoxon signed-rank test to ensure that the differences in perplexity are statistically significant. In the following sections we compare the performance of the different training methods and discuss the obtained perplexity results.

#### 4.1.1. Improvements from uncertainty in non-adaptation limited data setups

We first compare the perplexities obtained by the LSTM-RNN LMs trained only on the in-domain datasets, i.e. only VM or AMI as presented by the last column of Table 2 and Table 3. We can observe that the sampling+KNN method for learning from ASR confusion networks results in the lowest perplexity within each horizontal half, and this perplexity is significantly lower than LSTM-RNN LMs trained on the ASR 1-best hypothesis with or without KNN. For instance when training without the labeled training set, the relative reduction in perplexity on the VM test set is 9% (i.e. from 68.4 to 62.0) and that on the AMI test set is 12% (i.e. from 137.9 to 121.8).

Training based on the HMM formulation results in higher perplexities compared to the other models trained on ASR confusion networks or ASR 1-best. We will show below that the performance of this method improves by training along with the labeled training set and also in the adaptation setting.

Table 2: Perplexities obtained on the VM dataset by LSTM-RNN LMs using different training methods. LSTM 128-d for SWB+VM amd LSTM 64-d for VM only. Bold font denotes the lowest perplexity within each horizontal half and underline indicates performance statistically similar to the lowest perplexity. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, cn: ASR confusion network, CE: cross entropy, KL: Kullback–Leibler divergence, +KNN: bigram Kneser-Ney noising along with the method in the previous row.)

<table>
<tr><th colspan="2" rowspan="2">VM training set</th><th rowspan="2">Training method</th><th colspan="2">SWB+VM pre-trained</th><th colspan="2">SWB+VM adapted to VM</th><th colspan="2">Trained on VM only</th></tr>
<tr><th>Dev</th><th>Test</th><th>Dev</th><th>Test</th><th>Dev</th><th>Test</th></tr>
<tr><td rowspan="8">without labeled training set</td><td rowspan="2">unlab-ref</td><td>CE</td><td rowspan="2">68.5</td><td rowspan="2">70.7</td><td>40.4</td><td>43.0</td><td>48.2</td><td>52.1</td></tr>
<tr><td>+ KNN</td><td>40.7</td><td>43.7</td><td>47.5</td><td>50.2</td></tr>
<tr><td rowspan="2">unlab-1b</td><td>CE</td><td rowspan="6">84.7</td><td rowspan="6">83.8</td><td><u>52.4</u></td><td><u>56.0</u></td><td>66.3</td><td>72.7</td></tr>
<tr><td>+ KNN</td><td>54.6</td><td>59.6</td><td>62.9</td><td>68.4</td></tr>
<tr><td rowspan="4">unlab-cn</td><td>KL</td><td>51.2</td><td>55.0</td><td><u>58.4</u></td><td><u>63.7</u></td></tr>
<tr><td>HMM</td><td>53.5</td><td>57.3</td><td>74.0</td><td>81.9</td></tr>
<tr><td>sample</td><td><u>53.0</u></td><td><u>56.0</u></td><td><u>58.9</u></td><td>64.1</td></tr>
<tr><td>+ KNN</td><td>53.1</td><td><u>56.0</u></td><td>57.5</td><td>62.0</td></tr>
<tr><td rowspan="8">with labeled training set</td><td rowspan="2">lab-ref + unlab-ref</td><td>CE</td><td rowspan="2">69.6</td><td rowspan="2">71.1</td><td>40.5</td><td>42.8</td><td>48.7</td><td>51.9</td></tr>
<tr><td>+ KNN</td><td>40.0</td><td>42.1</td><td>44.8</td><td>47.3</td></tr>
<tr><td rowspan="2">lab-ref + unlab-1b</td><td>CE</td><td rowspan="6">78.2</td><td rowspan="6">77.1</td><td><u>47.8</u></td><td><u>50.6</u></td><td>58.4</td><td>62.7</td></tr>
<tr><td>+ KNN</td><td>49.1</td><td><u>51.6</u></td><td>56.0</td><td>59.0</td></tr>
<tr><td rowspan="4">lab-ref + unlab-cn</td><td>KL</td><td>47.4</td><td>50.2</td><td>55.2</td><td>59.2</td></tr>
<tr><td>HMM</td><td><u>48.1</u></td><td><u>50.8</u></td><td>60.6</td><td>64.5</td></tr>
<tr><td>sample</td><td><u>48.0</u></td><td><u>50.8</u></td><td><u>54.7</u></td><td>58.4</td></tr>
<tr><td>+ KNN</td><td><u>48.8</u></td><td><u>51.6</u></td><td>53.5</td><td>56.2</td></tr>
</table>

Table 3: Perplexities obtained on the AMI dataset by LSTM-RNN LMs using different training methods. LSTM 128-d for SWB+AMI amd LSTM 64-d for AMI only. Bold font denotes the lowest perplexity within each horizontal half and underline indicates performance statistically similar to the lowest perplexity. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, cn: ASR confusion network, CE: cross entropy, KL: Kullback–Leibler divergence, +KNN: bigram Kneser-Ney noising along with the method in the previous row)

| AMI training set | | Training method | SWB+AMI pre-trained | | SWB+AMI adapted to AMI | | Trained on AMI only | |
|---|---|---|---|---|---|---|---|---|
| | | | Dev | Test | Dev | Test | Dev | Test |
| without labeled training set | unlab-ref | CE | 88.9 | 97.1 | 61.9 | 68.3 | 72.4 | 81.9 |
| | | + KNN | | | 62.4 | 69.6 | 70.7 | 79.1 |
| | unlab-1b | CE | 111.5 | 124.7 | 89.5 | 102.6 | 119.5 | 144.9 |
| | | + KNN | | | 96.8 | 113.2 | 114.9 | 137.9 |
| | unlab-cn | KL | | | **87.2** | **100.2** | 109.6 | 130.0 |
| | | HMM | | | 97.9 | 113.1 | 139.0 | 165.8 |
| | | sample | | | 89.9 | 104.0 | 104.8 | 124.9 |
| | | + KNN | | | 92.1 | 107.3 | **102.2** | **121.8** |
| with labeled training set | lab-ref + unlab-ref | CE | 87.7 | 94.9 | 59.7 | 65.9 | 68.9 | 76.9 |
| | | + KNN | | | 60.3 | 66.4 | 67.7 | 75.4 |
| | lab-ref + unlab-1b | CE | 96.5 | 104.2 | 70.5 | 77.9 | 81.0 | 91.4 |
| | | + KNN | | | 75.6 | 85.0 | 80.6 | 90.4 |
| | lab-ref + unlab-cn | KL | | | 70.3 | 78.4 | 83.2 | 96.4 |
| | | HMM | | | 74.3 | 82.7 | 88.3 | 99.5 |
| | | sample | | | 70.6 | 79.9 | 77.9 | 87.9 |
| | | + KNN | | | **69.7** | **77.5** | **76.5** | **85.5** |

Table 4: Perplexities obtained on the AMI dataset by LSTM-RNN LMs trained on ASR hypotheses of labeled and unlabeled training sets using different methods. LSTM 128-d for SWB+AMI amd LSTM 64-d for AMI only. Bold font denotes the lowest perplexity and underline indicates performance statistically similar to the lowest perplexity. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, cn: ASR confusion network, CE: cross entropy, KL: Kullback–Leibler divergence, +KNN: bigram Kneser-Ney noising along with the method in the previous row)

| AMI training set | | Training method | SWB+AMI pre-trained | | SWB+AMI adapted to AMI | | Trained on AMI only | |
|---|---|---|---|---|---|---|---|---|
| | | | Dev | Test | Dev | Test | Dev | Test |
| with labeled training set (labels unused) | lab-1b + unlab-1b | CE | 111.0 | 123.6 | <u>88.5</u> | 101.8 | 112.4 | 134.3 |
| | | + KNN | | | 96.4 | 112.7 | 109.3 | 130.5 |
| | lab-cn + unlab-cn | KL | | | **86.5** | **99.3** | 105.3 | 124.8 |
| | | HMM | | | 97.0 | 112.0 | 134.4 | 159.3 |
| | | sample | | | 89.3 | 103.9 | 101.4 | 119.8 |
| | | + KNN | | | 90.8 | 105.8 | **97.2** | **114.7** |

*4.1.2. Improvements brought by labeled training data*

Comparison of the LSTM-RNN LMs trained with vs. without the labeled training data, using only the in-domain data (i.e. only VM or AMI), shows that a small amount of manually transcribed in-domain data results in a significant reduction in perplexity for all the training methods. However, this comparison is biased as the amount of training data is different in the two settings. In order to evaluate the improvement brought by manual labeling independently of the amount of data, we trained LSTM-RNN LMs on ASR 1-best hypotheses or confusion networks of both labeled and unlabeled training sets. The resulting perplexities are presented in the Table 4. Comparison of the results in Table 3, i.e., without using the manual transcriptions of the labeled set, and Table 4 helps us to confirm that greater perplexity reductions are obtained due to the manual transcriptions of the labeled training data. The increase in the amount of unlabeled training data, in the form of ASR 1-best or confusion networks of the labeled training set, results in significant but smaller perplexity reductions.

Having verified that adding a small amount of labeled in-domain training data can lead to significant reductions in perplexity, we would like to highlight the performance of the different training methods. On the VM test set, the best performing sampling+KNN method shows a perplexity reduction of 9% relative (from 62.0 to 56.2). On the AMI test set, the best performing sampling+KNN method shows 30% relative reduction in perplexity (from 121.8 to 85.5). The availability of the labeled train set also leads to significant perplexity reductions for the KL method as well as for training on ASR 1-best hypotheses. It should be noted that training on the in-domain datasets using the HMM formulation by including the labeled training data achieves 21% and 40% relative reductions in perplexity on the VM and AMI test sets, respectively, as compared to training without the labeled training data.

#### *4.1.3. Adaptation setting*

Table 2 and Table 3 also highlight the differences in the performance of the models trained only on the in-domain data vs. those adapted after pre-training on a combination of SWB and the in-domain data. The adapted models clearly outperform their in-domain only counterparts in all cases, both on the VM and AMI datasets. However, it must be noted that such an out-of-domain dataset, consisting of manual transcriptions of about 300 hours of spoken conversations, may not be available for most languages. Moreover, the evaluation reveals some useful insights on the presented methods for training RNN LMs on ASR confusion networks.

Adapting the LSTM-RNN LMs to the in-domain ASR hypotheses, without the labeled training data, achieves the lowest perplexities with the KL method. However, adaptation along with the labeled training data results in similar perplexities when training with ASR 1-best transcripts or ASR confusion networks, in case of the KL divergence and sampling based methods. Adaptation of the LSTM-RNN LMs using the HMM formulation results in much lower perplexities as compared to HMM based training only on the in-domain data. The perplexity reductions are observed with and without the labeled training data.

Moreover, we also observe that applying Kneser-Ney noising (KNN) when adapting to the in-domain data can lead to increased perplexities. This is more evident on the AMI dataset, both with and without the labeled training data.

#### *4.1.4. Improvements from ASR confusion networks: n-gram versus RNN LMs*

We also evaluate the performance of 3-gram LMs trained on ASR 1-best hypotheses and ASR confusion networks, and compare the resulting reductions in perplexity to those obtained from the LSTM-RNN LMs. 3-gram LMs are trained on ASR 1-best hypotheses or manual transcriptions or a combination of both using modified interpolated KN smoothing (Chen and Goodman, 1996). Classical KN smoothing cannot be applied directly to ASR confusion networks as words/arcs carry fractional weights or scores. A modified interpolated *expected* KN smoothing (ieKN) approach has been proposed in the literature to handle such fractional counts (Zhang and Chiang, 2014). This has been applied to learn n-gram LMs from crowdsourced and ASR transcriptions (Levit et al., 2018), and shown to result in n-gram LMs with lower perplexities as compared to other previous works.

We extend the ieKN approach to train n-gram LMs on ASR confusion networks. Our approach first extracts n-gram bin sequences from the confusion network and then populates different possible $n$-th (i.e. highest) order word sequences. Each $n$-th order word sequence is assigned a score by multiplying the associated arc posteriors with each other. The ieKN smoothing approach is applied to obtain the $n$-th order probability estimates. Then the recursive smoothing, analogous to standard KN smoothing, is applied to obtain the lower order probability estimates (Zhang and Chiang, 2014).

Table 5 presents the perplexities obtained by 3-gram and LSTM-RNN LMs on the VM and AMI datasets. LSTM-RNN LM perplexities correspond to the CE+KNN and sample+KNN methods in Table 2 and Table 3. Firstly, we can observe that training 3-gram LMs on ASR confusion networks with the ieKN

Table 5: Perplexities of 3-gram LMs and LSTM-RNN LMs trained on ASR 1-best hypotheses and ASR confusion networks. 3-gram and LSTM-RNN LMs trained on confusion networks using ieKN smoothing and sample+KNN methods respectively. KN smoothing and CE+KNN otherwise. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, cn: ASR confusion network. Note that the ASpIRE LM with a larger vocabulary is used to decode AMI Dev and Test sets, and the corresponding perplexities are not comparable.)

| Training set / LM | | 3-gram LM | | | | LSTM-RNN LM | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | VM | | AMI | | VM | | AMI | |
| | | Dev | Test | Dev | Test | Dev | Test | Dev | Test |
| | decode LM | 77.3 | 78.0 | 142.3 | 163.6 | - | - | - | - |
| without labeled training set | unlab-ref | 52.0 | 54.6 | 69.5 | 76.6 | 47.5 | 50.2 | 70.7 | 79.1 |
| | unlab-1b | 68.2 | 72.8 | 100.1 | 116.5 | 62.9 | 68.4 | 114.9 | 137.9 |
| | unlab-cn | **64.1** | **68.5** | **95.9** | **111.0** | **57.5** | **62.0** | **102.2** | **121.8** |
| with labeled training set | lab-ref + unlab-ref | 50.1 | 52.5 | 67.8 | 74.5 | 44.8 | 47.3 | 67.7 | 75.4 |
| | lab-ref + unlab-1b | 64.0 | 67.4 | 80.5 | 89.8 | 56.0 | 59.0 | 80.6 | 90.4 |
| | lab-ref + unlab-cn | **61.2** | **64.3** | **78.6** | **87.4** | **53.5** | **56.2** | **76.5** | **85.5** |

based approach results in a significant reduction in perplexity as compared to 3-gram LMs trained on ASR 1-best hypotheses. A comparison of perplexities across 3-gram and LSTM-RNN LMs shows that the LSTM-RNN LMs achieve lower perplexities on the VM dataset, namely 62.0 and 56.2 on the VM test set, without and with the labeled training set, versus 68.5 and 64.3 obtained by the ieKN 3-gram LM, respectively. However, ieKN 3-gram LMs obtain lower perplexities than LSTM-RNN LMs on the AMI dataset. This could be due to the fact that the VM dataset has longer utterances (i.e. speaker turns) as compared to the AMI dataset. Moreover, we can also observe that perplexity reductions obtained by the use of ASR confusion networks instead of ASR 1-

best hypotheses are greater when the labeled training data was not available, with a few exceptions.

### 4.2. ASR evaluation

We further evaluate the best performing LSTM-RNN LMs for their effectiveness on improving the ASR performance. Typically, LSTM-RNN LMs are used to perform rescoring of lattices or n-best lists decoded using n-gram LMs. In our experimental setup, lattice rescoring cannot be applied on the AMI dataset due to a mismatch between the vocabulary of the ASpIRE model used for decoding and that of the trained LSTM-RNN LMs. In favour of consistency, we perform n-best list rescoring on both the VM and AMI datasets. The matched pairs sentence-segment word error test (Gillick and Cox, 1989), from the NIST scoring toolkit[7], is used to ensure that the differences in WER are statistically significant. The results and discussion are divided into the following sections.

#### 4.2.1. LSTM-RNN LMs trained only on in-domain data

Table 6 presents the WER obtained by the LSTM-RNN LMs trained on the ASR 1-best hypotheses and confusion networks, along with the labeled training set. ASR lattices were decoded using the seed LMs for the VM and AMI datasets, as presented in Section 3.3.3. These lattices were rescored using a KN smoothed 3-gram LM trained on a combination of the labeled data and 1-best transcripts of the unlabeled data, denoted as 'lab ref+unlab 1b' in Table 6. 100-best lists were then obtained from these n-gram rescored lattices. Finally, the 100-best lists were rescored using the LSTM-RNN LMs such that the original AM scores are retained and the 3-gram and LSTM-RNN LM scores are linearly interpolated. This is denoted as '3g+LSTM' in Table 6. The weight for linear interpolation was tuned on the development set.

The first observation from Table 6 is that the room left for WER reduction, i.e. the difference between 'lab ref+unlab 1b' 3g LM rescoring and 'lab

[7] https://github.com/usnistgov/SCTK

Table 6: WER after rescoring 100-best lists obtained from (lab ref + unlab 1b) 3-gram LM rescored lattices. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, cn: ASR confusion network, KN: modified interpolated Kneser-Ney smoothing, CE: cross entropy, KL: Kullback–Leibler divergence, KNN: bigram Kneser-Ney noising. Bold font denotes the lowest WER and underline indicates performance statistically similar to the lowest WER.)

| LM configuration | | | | VM | | AMI | |
|---|---|---|---|---|---|---|---|
| Training set / LM | | Type | Method | Dev | Test | Dev | Test |
| | decode LM | 3g | KN | 39.52 | 39.77 | 32.27 | 35.12 |
| with labeled training set | lab + unlab ref | 3g+LSTM | CE + KNN | 35.22 | 35.32 | 31.45 | 34.00 |
| with labeled training set | lab ref + unlab 1b | 3g | KN | 37.42 | 37.93 | 34.01 | 36.70 |
| with labeled training set | lab ref + unlab 1b | 3g+LSTM | CE + KNN | <u>36.42</u> | 36.69 | **33.18** | <u>35.85</u> |
| with labeled training set | lab ref + unlab cn | 3g+LSTM | KL | 36.56 | 36.75 | <u>33.24</u> | <u>35.83</u> |
| with labeled training set | lab ref + unlab cn | 3g+LSTM | sample + KNN | **36.14** | **36.33** | <u>33.25</u> | **35.77** |

ref+unlab ref' 3g+LSTM LM rescoring, is small. The absolute difference is about 2.6% on both the VM and AMI test sets. This implies that WER reduction through RNN LMs is a difficult task in such limited training data setups. On the VM dataset, the LSTM-RNN LM trained on the ASR confusion networks using sampling+KNN results in the lowest WER. On the VM dev set, the difference is statistically significant (at p=0.05) compared to the KL method. On the VM test set, the difference is statistically significant compared to the KL method as well as CE+KNN on 'lab ref+unlab 1b'. On the AMI dataset, the WERs obtained by the LSTM-RNN LMs trained on ASR 1-best hypotheses and confusion networks are nearly the same. We also observe that rescoring the ASpIRE model decoded lattices with the in-domain 'lab ref+unlab 1b' 3g LM increases the WER. In the following section we discuss the AMI WER performance without the in-domain 'lab ref+unlab 1b' 3g LM rescoring.

#### *4.2.2. AMI WER evaluation with and without labeled training data*

The AMI setup in our experiments allows us to evaluate the WER in two settings, first wherein additional labeled training data is available and second wherein reference transcriptions or ASR hypotheses of the labeled data may be considered for training the LSTM-RNN LMs. In Section 4.2.1 we observed that rescoring with the in-domain 3g LM resulted in a WER increase on AMI. To perform a WER evaluation without this bias, 100-best lists were directly obtained from the ASpIRE model decoded lattices and were rescored using the LSTM-RNN LMs. The resulting WERs are presented in Table 7.

The first point to observe is that rescoring 100-best lists obtained from the ASpIRE model reduces the WER below the first pass decoding results, unlike the AMI results in Table 6. Second, the availability of labeled training data results in small but consistent reductions in WER. The WERs obtained by the LSTM-RNN LMs trained on the ASR 1-best hypotheses and confusion networks are nearly the same, both with and without the labeled training set. However, it must be noted that the gap between WERs of the first pass decoding ('decode LM') and rescoring with LSTM-RNN LMs trained on labeled and unlabeled data ('lab-ref + unlab-ref') is only partly filled. This motivates the need for better methods for training RNN LMs from uncertain ASR hypotheses.

#### *4.2.3. WER performance in the adaptation setting*

Perplexity evaluation in Table 3 and Table 2 showed that LSTM-RNN LMs pre-trained on SWB and adapted to the in-domain data resulted in the lowest perplexities. Table 8 presents the WER obtained by the adapted LSTM-RNN LMs. We present the WER of LMs trained with standard CE criteria and only on 1-best ASR hypotheses as their perplexities are as good as the other compared methods. 100-best lists for AMI are obtained from ASpIRE model decoded lattices. 100-best lists for VM are obtained after 3g LM rescoring. The VM WERs in Table 8 can be compared to those in Table 6 and the AMI WERs in Table 8 can be compared to those in Table 7.

We can observe that the adaptation setting results in small, but significant,

Table 7: WER on the AMI dataset after rescoring 100-best lists obtained from the ASpIRE model using LSTM-RNN LMs trained with and without labeled training set. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, cn: ASR confusion network, KN: modified interpolated Kneser-Ney smoothing, CE: cross entropy, KL: Kullback–Leibler divergence, KNN: bigram Kneser-Ney noising. Bold font denotes the lowest WER within each horizontal third and underline indicates performance statistically similar to the lowest WER.)

| LM configuration | | | | AMI | |
|---|---|---|---|---|---|
| Training set / LM | | Type | Training method | Dev | Test |
| | decode LM | 3g | KN | 32.27 | 35.12 |
| without labeled training set | unlab ref | 3g+LSTM | CE + KNN | 30.19 | 32.59 |
| without labeled training set | unlab-1b | 3g+LSTM | CE + KNN | 31.80 | 34.36 |
| without labeled training set | unlab-cn | 3g+LSTM | KL | 31.77 | 34.48 |
| without labeled training set | unlab-cn | 3g+LSTM | sample + KNN | **31.71** | **34.28** |
| with labeled training set (labels used) | lab-ref + unlab-ref | 3g+LSTM | CE + KNN | 29.93 | 32.44 |
| with labeled training set (labels used) | lab-ref + unlab-1b | 3g+LSTM | CE + KNN | 31.44 | 34.16 |
| with labeled training set (labels used) | lab-ref + unlab-cn | 3g+LSTM | KL | 31.44 | 34.16 |
| with labeled training set (labels used) | lab-ref + unlab-cn | 3g+LSTM | sample + KNN | **31.42** | **34.13** |
| with labeled training set (labels unused) | lab-1b + unlab-1b | 3g+LSTM | CE + KNN | 31.68 | 34.27 |
| with labeled training set (labels unused) | lab-cn + unlab-cn | 3g+LSTM | KL | 31.67 | **34.26** |
| with labeled training set (labels unused) | lab-cn + unlab-cn | 3g+LSTM | sample + KNN | **31.62** | 34.31 |

WER reductions compared to training only on in-domain data. In the case of VM, the best performing LSTM-RNN LM trained only on in-domain data results in 3.4% and 4.2% relative WER reduction on the dev and test sets, respectively, while the adapted LSTM-RNN LM results in 5.6% and 6.0% relative WER reduction on the dev and test sets, respectively. In the case of AMI, the best performing LSTM-RNN trained only on in-domain data results in 2.6% and

Table 8: WER evaluation in the adaptation setting. LSTM-RNN LM rescoring on 100-best lists obtained from (lab ref + unlab 1b) 3g LM rescored lattices for VM. LSTM-RNN LM rescoring on 100-best lists obtained from the ASpIRE model. (lab: labeled training set, unlab: unlabeled training set, ref: manual transcriptions, 1b: ASR 1-best, KN: modified interpolated Kneser-Ney smoothing, CE: cross entropy.)

| LM configuration | | | | VM | | AMI | |
|---|---|---|---|---|---|---|---|
| Training set / LM | | Type | Method | Dev | Test | Dev | Test |
| | decode LM | 3g | KN | 39.52 | 39.77 | 32.27 | 35.12 |
| with labeled training set | lab + unlab ref | 3g+LSTM | CE | 34.65 | 34.74 | 29.45 | 32.05 |
| | lab ref + unlab 1b | 3g | KN | 37.42 | 37.93 | - | - |
| | | 3g+LSTM | CE | 35.33 | 35.63 | 31.10 | 33.75 |

2.8% relative WER reduction on the dev and test sets, respectively, while the adapted LSTM-RNN LM results in 3.6% and 3.9% relative WER reduction on the dev and test sets, respectively. It should also be noted that the differences in the WER from training versus adaptation of LSTM-RNN LMs on reference transcriptions is quite small. We can state that, in the absence of a large relevant corpus for pre-training LSTM-RNN LMs, exploiting the ASR hypotheses from in-domain speech data with more effective training methods can lead to a better ASR performance.

## 5. Conclusion

We explored three different methods to train and adapt RNN LMs on ASR confusion networks obtained from unlabeled in-domain speech, with the aim of exploiting uncertainty in ASR transcriptions, while targeting limited training data scenarios. Overall, the method based on sampling of paths from the ASR confusion networks, as well as the method which minimizes KL divergence between the model predictions and confusion bin posteriors, lead to statistically significant reductions in perplexity, as compared to training on ASR 1-best hypotheses. Training based on the HMM formulation resulted in higher perplexities as compared to training on ASR 1-best hypotheses. However, evaluation of perplexities in the adaptation settings, wherein the RNN LM was pre-trained on out-of-domain conversations, shows that the three methods perform similarly to training on 1-best ASR hypotheses.

ASR evaluation based on rescoring of n-best lists showed that training RNN LMs on ASR confusion networks results in small but significant reductions in WER on the VM dataset, as compared to RNN LMs trained on ASR 1-best hypotheses. ASR evaluation on the AMI datatset did not result into any WER reduction. ASR evaluation of the pre-trained RNN LMs adapted to the manual transcriptions of entire in-domain data reveals that similar WER reductions could be achieved by training only on the in-domain data. This motivates the need for more effective methods to train RNN LMs on uncertain ASR hypotheses.

Apart from discovering more effective methods to train RNN LMs from in-domain speech, we also envisage to explore the recent Transformer LMs for this task. Incorporating the alternate hypotheses and uncertainties using the self attention mechanism of Transformers seems to be an interesting direction for future work. However, training and adaptation with limited in-domain data will remain as an interesting challenge in this direction.

## 6. Acknowledgments

This work was supported by the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 825081 COMPRISE (`https://www.compriseh2020.eu/`). Experiments were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see `https://www.grid5000.fr`).

## References

Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F., Weber, G., 2020. Common voice: A massivelymultilingual speech corpus, in: Proceedings of the 12th Language Resources and Evaluation Conference, pp. 4218–4222.

Bacchiani, M., Riley, M., Roark, B., Sproat, R., 2006. MAP adaptation of stochastic grammars. Computer Speech and Language 20, 41–68.

Bacchiani, M., Roark, B., 2003. Unsupervised language model adaptation, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 224–227.

Ballinger, B., Allauzen, C., Gruenstein, A., Schalkwyk, J., 2010. On-demand language model interpolation for mobile speech input, in: Proceedings of Interspeech, pp. 1812–1815.

Bellegarda, J.R., 2004. Statistical language model adaptation: Review and perspectives. Speech Communication 42, 93–108.

Burger, S., Weilhammer, K., Schiel, F., Tillmann, H.G., 2000. Verbmobil data collection and annotation, in: Verbmobil: Foundations of Speech-to-Speech Translation, pp. 537–549.

Chen, S.F., Goodman, J., 1996. An empirical study of smoothing techniques for language modeling, in: Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 310–318.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation, in: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724–1734.

Deena, S., Hasan, M., Doulaty, M., Saz, O., Hain, T., 2016. Combining feature and model-based adaptation of RNNLMs for multi-genre broadcast speech recognition, in: Proceedings of Interspeech, pp. 2343–2347.

Dror, R., Baumer, G., Shlomov, S., Reichart, R., 2018. The hitchhiker's guide to testing statistical significance in natural language processing, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1383–1392.

Gales, M., Young, S., 2007. The application of hidden Markov models in speech recognition. Foundations and Trends in Signal Processing 1, 195–304.

Gangireddy, S.R., Swietojanski, P., Bell, P., Renals, S., 2016. Unsupervised adaptation of recurrent neural network language models, in: Proceedings of Interspeech, pp. 2333–2337.

Gillick, L., Cox, S., 1989. Some statistical issues in the comparison of speech recognition algorithms, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 532–535.

Godfrey, J., Holliman, E., McDaniel, J., 1992. Switchboard: telephone speech corpus for research and development, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 517–520.

Hakkani-Tür, D., Béchet, F., Riccardi, G., Tur, G., 2006. Beyond ASR 1-best: Using word confusion networks in spoken language understanding. Computer Speech and Language 20, 495–514.

Haznedaroglu, A., Arslan, L.M., 2014. Language model adaptation for automatic call transcription, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4102–4106.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Computation. 9, 1735–1780.

Huang, C., Chen, Y., 2020. Learning spoken language representations with neural lattice language modeling, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 3764–3769.

Huang, R., Li, K., Arora, A., Povey, D., Khudanpur, S., 2020. Efficient MDI adaptation for n-gram language models, in: Proceedings of Interspeech, pp. 4916–4920.

Inan, H., Khosravi, K., Socher, R., 2017. Tying word vectors and word classifiers: A loss framework for language modeling, in: Proceedings of International Conference on Learning Representations (ICLR).

Irie, K., 2020. Advancing neural language modeling in automatic speech recognition. Ph.D. thesis. RWTH Aachen University.

Irie, K., Kumar, S., Nirschl, M., Liao, H., 2018. RADMM: Recurrent adaptive mixture model with applications to domain robust language modeling, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6079–6083.

Jagfeld, G., Vu, N.T., 2017. Encoding word confusion networks with recurrent neural networks for dialog state tracking, in: Proceedings of the Workshop on Speech-Centric Natural Language Processing, pp. 10–17.

Kim, J.Y., Liu, C., Calvo, R.A., McCabe, K., Taylor, S.C.R., Schuller, B.W., Wu, K., 2019. A comparison of online automatic speech recognition systems and the nonverbal responses to unintelligible speech. arXiv preprint arXiv:1904.12403 .

Kuznetsov, V., Liao, H., Mohri, M., Riley, M., Roark, B., 2016. Learning n-gram language models from uncertain data, in: Proceedings of Interspeech, pp. 2323–2327.

Ladhak, F., Gandhe, A., Dreyer, M., Mathias, L., Rastrow, A., Hoffmeister, B., 2016. LatticeRNN: Recurrent neural networks over lattices, in: Proceedings of Interspeech, pp. 695–699.

Langzhou Chen, Gauvain, J.., Lamel, L., Adda, G., 2003. Unsupervised language model adaptation for broadcast news, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 220–223.

Lecorvé, G., Dines, J., Hain, T., Motlicek, P., 2012. Supervised and unsupervised web-based language model domain adaptation, in: Proceedings of Interspeech, pp. 182–185.

Levit, M., Parthasarathy, S., Chang, S., 2018. What to expect from expected Kneser-Ney smoothing, in: Proceedings of Interspeech, pp. 3378–3382.

Li, K., Xu, H., Wang, Y., Povey, D., Khudanpur, S., 2018. Recurrent neural network language model adaptation for conversational speech recognition, in: Proceedings of Interspeech, pp. 3373–3377.

Ma, M., Nirschl, M., Biadsy, F., Kumar, S., 2017. Approaches for neural-network language model adaptation, in: Proceedings of Interspeech, pp. 259–263.

Matusov, E., Kanthak, S., Ney, H., 2005. On the integration of speech recognition and statistical machine translation, in: Proceedings of Interspeech, pp. 3177–3180.

Meng, S., Thambiratnam, K., Lin, Y., Wang, L., Li, G., Seide, F., 2010. Vocabulary and language model adaptation using just one speech file, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5410–5413.

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., Khudanpur, S., 2010. Recurrent neural network based language model., in: Proceedings of Interspeech, pp. 1045–1048.

Niesler, T., Willett, D., 2002. Unsupervised language model adaptation for lecture speech transcription, in: Proceedings of International Conference on Spoken Language Processing (ICSLP), pp. 1413–1416.

Ozerov, A., Lagrange, M., Vincent, E., 2013. Uncertainty-based learning of acoustic models from noisy data. Computer Speech and Language 27, 874–894.

Pal, V., Guillot, F., Shrivastava, M., Renders, J., Besacier, L., 2020. Modeling ASR ambiguity for neural dialogue state tracking, in: Proceedings of Interspeech, pp. 1545–1549.

Povey, D., Peddinti, V., Galvez, D., Ghahremani, P., Manohar, V., Na, X., Wang, Y., Khudanpur, S., 2016. Purely sequence-trained neural networks for ASR based on lattice-free MMI, in: Proceedings of Interspeech, pp. 2751–2755.

Pratap, V., Xu, Q., Sriram, A., Synnaeve, G., Collobert, R., 2020. MLS: A large-scale multilingual dataset for speech research, in: Proceedings of Interspeech, pp. 2757–2761.

Press, O., Wolf, L., 2017. Using the output embedding to improve language models, in: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL): Volume 2, Short Papers, pp. 157–163.

Pusateri, E., Gysel, C.V., Botros, R., Badaskar, S., Hannemann, M., Oualil, Y., Oparin, I., 2019. Connecting and comparing language model interpolation techniques, in: Proceedings of Interspeech, pp. 3500–3504.

Rabiner, L., Juang, B., 1986. An introduction to hidden Markov models. IEEE ASSP Magazine 3, 4–16.

Raju, A., Hedayatnia, B., Liu, L., Gandhe, A., Khatri, C., Metallinou, A., Venkatesh, A., Rastrow, A., 2018. Contextual language model adaptation for conversational agents, in: Proceedings of Interspeech, pp. 3333–3337.

Renals, S., Hain, T., Bourlard, H., 2007. Recognition and understanding of meetings the AMI and AMIDA projects, in: Proceedings of IEEE Workshop on Automatic Speech Recognition Understanding (ASRU), pp. 238–247.

Rizk, B., 2019. Evaluation of state of art open-source ASR engines with local inferencing. Bachelor's thesis. Institute of Information Systems, Hof University.

Sperber, M., Neubig, G., Niehues, J., Waibel, A., 2017. Neural lattice-to-sequence models for uncertain inputs, in: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1380–1389.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15, 1929–1958.

Sundermeyer, M., Ney, H., Schlüter, R., 2015. From feedforward to recurrent LSTM neural networks for language modeling. IEEE/ACM Transactions on Audio, Speech, and Language Processing 23, 517–529.

Tam, Y., Schultz, T., 2009. Correlated bigram LSA for unsupervised language model adaptation, in: Proceedings of Advances in Neural Information Processing Systems (NIPS), p. 1633–1640.

Toshniwal, S., Kannan, A., Chiu, C., Wu, Y., Sainath, T.N., Livescu, K., 2018. A comparison of techniques for language model integration in encoder-decoder speech recognition, in: Proceedings of IEEE Spoken Language Technology (SLT) Workshop, pp. 369–375.

Tur, G., Stolcke, A., 2007. Unsupervised language model adaptation for meeting recognition, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 173–176.

Xie, S., Chen, L., 2013. Evaluating unsupervised language model adaptation methods for speaking assessment, in: Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications, pp. 288–292.

Xie, Z., Wang, S.I., Li, J., Lévy, D., Nie, A., Jurafsky, D., Ng, A.Y., 2017. Data noising as smoothing in neural network language models, in: Proceedings of International Conference on Learning Representations (ICLR).

Xu, H., Povey, D., Mangu, L., Zhu, J., 2011. Minimum Bayes risk decoding and system combination based on a recursion for edit distance. Computer Speech and Language 25, 802–828.

Xu, P., Karakos, D., Khudanpur, S., 2009. Self-supervised discriminative training of statistical language models, in: Proceedings of IEEE Workshop on Automatic Speech Recognition Understanding (ASRU), pp. 317–322.

Yang, X., Liu, J., 2015. Using word confusion networks for slot filling in spoken language understanding, in: Proceedings of Interspeech, pp. 1353–1357.

Zhang, H., Chiang, D., 2014. Kneser-Ney smoothing on expected counts, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers), pp. 765–774.

Zhang, R., Kikui, G., 2006. Integration of speech recognition and machine translation: Speech recognition word lattice translation. Speech Communication 48, 321–334.

Çelebi, A., Sak, H., Dikici, E., Saraçlar, M., Lehr, M., Prud'hommeaux, E., Xu, P., Glenn, N., Karakos, D., Khudanpur, S., Roark, B., Sagae, K., Shafran, I., Bikel, D., Callison-Burch, C., Cao, Y., Hall, K., Hasler, E., Koehn, P., Lopez, A., Post, M., Riley, D., 2012. Semi-supervised discriminative language modeling for Turkish ASR, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5025–5028.