PERSONALIZATION STRATEGIES FOR END-TO-END SPEECH RECOGNITION SYSTEMS

Aditya Gourav* Linda Liu* Ankur Gandhe Yile Gu
Guitang Lan Xiangyang Huang Shashank Kalmane Gautam Tiwari
Denis Filimonov Ariya Rastrow Andreas Stolcke Ivan Bulyko

Amazon Alexa

ABSTRACT

The recognition of personalized content, such as contact names, remains a challenging problem for end-to-end speech recognition systems. In this work, we demonstrate how first- and second-pass rescoring strategies can be leveraged together to improve the recognition of such words. Following previous work, we use a shallow fusion approach to bias towards recognition of personalized content in the first-pass decoding. We show that such an approach can improve personalized content recognition by up to 16% with minimum degradation on the general use case. We describe a fast and scalable algorithm that enables our biasing models to remain at the word-level, while applying the biasing at the subword level. This has the advantage of not requiring the biasing models to be dependent on any subword symbol table. We also describe a novel second-pass de-biasing approach: used in conjunction with a first-pass shallow fusion that optimizes on oracle WER, we can achieve an additional 14% improvement on personalized content recognition, and even improve accuracy for the general use case by up to 2.5%.

Index Terms— language modeling, automatic speech recognition, rescoring, shallow fusion, personalization

1. INTRODUCTION

The successful recognition of personalized content, such as a user’s contacts or custom smart home device names, is essential for automatic speech recognition (ASR). Personalized content recognition is challenging as such words can be very rare or have low probability for the user population overall. For instance, a user’s contact list may contain foreign names or unique nicknames, and they may freely name their smart home devices.

This problem is exacerbated for end-to-end (E2E) systems, such as those based on CTC [1], LAS [2], or RNN-T [3]. Unlike hybrid ASR systems, which include acoustic and language model (LM) components that are trained separately, E2E systems use a single network that is trained end-to-end. Whereas in a hybrid system, the LM component can be trained separately on any written text, in an E2E system, the training is generally restricted to acoustic-text pairs. As a result, E2E systems are often trained with less data than hybrid systems, making personalized content recognition particularly challenging given the limited representation during training. Furthermore, hybrid systems are able to incorporate personal content into the decoding search graph, i.e., via class-based language models and on-the-fly composition of biasing phrases and n-grams [4][5][6][7][8].

Various approaches have been proposed for improving personalized content recognition for E2E models, including model fine-tuning with real or synthesized audio data [9], incorporating personalized content directly during E2E training using a separate bias-encoder module [10], using a token passing decoder with efficient token recombination during inference [11], and shallow fusion (e.g., [12][8][13][14]).

One popular approach to improve personalized content recognition is via shallow fusion [15]. In shallow fusion, the scores from an external language model $Score_{SF}(y)$ scaled by a factor $\lambda$ are combined with the main decoding scores $P_{RNNT}(y \mid x)$ during beam search:

$$\hat{y} = \arg\max_{y} \log P_{RNNT}(y \mid x) + \lambda \log Score_{SF}(y)$$  \hspace{1cm} (1)

This biasing can be applied at word boundaries [8], at the subword level $P_{RNNT}(y \mid x)$, or at the subword level $P_{RNNT}(y \mid x)$. Given that E2E models generally used a constrained beam [16], applying biasing only at word boundaries cannot improve performance if the relevant word does not already appear in the beam. As a result, compared to grapheme-level biasing which tends to keep the relevant words on the beam, word-level biasing results in less improvement on proper nouns such as contact names [10]. Applying biasing at the subword level, which would result in sparser matches at each step of the beam compared to the grapheme level, results in further improvements [13].

One challenge in applying biasing at the subword level, particularly for personalization, is that each of the biasing models needs to be built at the subword level and include all possible segmentations of a given word. This can be expensive when we have one or more models per user, particularly if the wordpiece model used to train the first-pass model often changes. Unlike previous work,
which generally relies on composition with a speller FST to transduce a sequence of wordpieces into the corresponding word (e.g., \[13\] [10][17]), we describe a novel prefix-matching algorithm in Section 3.2.2 that enables the language models to be kept at the word level and applies the subword decomposition at inference time.

Another challenge these shallow fusion approaches to personalization is how to improve recognition of personalized content while not degrading performance on general non-personalized content; to this end, several strategies for applying contextual biasing have been proposed \[13\] [14][11]. Many of these strategies reveal that applying shallow fusion in context minimizes, but does not completely remove, the degradation observed on general data, and do not discuss the potential impact of second-pass rescoring. For example, \[14\] finds that applying contextual shallow fusion decreases the negative impact on general content while maintaining performance on the shallow fusion content; however, even in this best case, they report a slight degradation of 5.8% (6.9 to 7.3 WER, cf. 12.5 for non-contextual shallow fusion) on general data.

On one hand, a more aggressive shallow fusion model enables more personalized content to appear in the n-best hypotheses but on the other hand, it is also more likely to cause false recognitions of the biased personalized content. To address this, we present a strategy in which we optimize shallow fusion for the n-best, as opposed to the 1-best, hypotheses, thereby maximizing the personalized content present in the n-best. To recover the correct 1-best, we explore a novel second-pass de-biasing approach that optimizes the combination of the E2E, shallow fusion, and second-pass scores.

3. METHODS

3.1. Baseline RNN-T model

Following \[13\], our baseline RNN-T model consists of an encoder comprised of five LSTM layers of size 1024, and a two-layer LSTM prediction network of size 1024 with an embedding layer of 512 units. The softmax layer consists of 4k (subword) output units. Our prediction network of size 1024 with an embedding layer of 512 units. The softmax layer consists of 4k (subword) output units. Our model was trained on over 200k hours of anonymized utterances from interactions with a voice assistant according to the minimum word error rate criterion \[19\] [18].

3.2. First-pass shallow fusion

3.2.1. Personalized models

For each anonymized user in our test set, we construct three personalized models, corresponding to (1) contact names (2) smart home device names and (3) enabled application names. Each of these models is represented as a word-level weighted finite state transducer (FST). An example is shown in Figure 1a. For simplicity, in our experiments, each word level arc has the same weight of -1. On average, each user has 600 personalized contact names, 50 device names, and 70 enabled applications.

3.2.2. Subword rescoring with lookahead

We describe our approach to biasing at the subword level using our word-level personalized models (Algorithm 1). We leverage ideas similar to \[20\] for subword level lookahead weight pushing and start with a word-level model represented as an FST, such as the one shown in Figure 1a. In this case, there are three paths associated with this FST, containing the words “play”, “player”, and “play-ground”. In Figure 1b, we show the subword breakdown for these words, based on some wordpiece model. The weights on each path are determined via Algorithm 1. Notice that the net weight for each path remains the same: i.e., the weight between state 0 and 5 (representing the word “player”) in the subword-level FST is (-1.6)+(-4.8) = -8, which is the same as the weight for the same word in the word-level FST. The weight \(w_{\text{pushed}}\) for each transition state is determined as follows: \(w_{\text{pushed}} = \frac{L}{N} (w_e \cdot w_{\text{lookahead}} - w_{\text{prev}})\), where \(L\) is the length of the prefix so far and \(N\) is the longest length of all matched words. In our example, given the input sequence “play” (pl, ay, ...) from Figure 1 we can see there are three arcs prefixed with the subword “pl”; thus, we have \(L = 2\), \(N = 10\), and the pushed weight is -8/2/10 = -1.6. Additionally, similar to \[10\] [17], we add fallback arcs for each non-final state with a weight equal to the negation of the current total weight up to that point.

This approach is beneficial as it avoids unnecessary arc expansion and provides a heuristic approach to perform subword-level rescoring without the need to build the biasing FST itself directly at the subword level. Additionally, this prefix matching approach enables us to consider any possible subword sequences for a word. To optimize the search for arcs that have a common prefix string, we sort the input arc in lexicographic order so that we can use binary search to find the lower and upper bound of arc indices. As we continue to process subword input, we are able to narrow down the search range quickly. We also cache all newly created states in subword level FST \(\mathcal{S}\), which results in efficient weight evaluation.

3.2.3. Contextual boosting model

Following previous work such as \[4\] [11][5][14], we construct a class-based language model containing three classes: contact names, home automation device names, and application names. To build the contextual biasing LM, we identified all utterances containing words that were annotated with aforementioned classes. We then
replaced the word(s) in the utterance with the corresponding class tag (e.g., @contactname). All utterances with the replaced class tags that occurred a minimum of 10 times were included in the contextual biasing FST. Unlike a typical class-based model, all arcs on the class-based model are unweighted. Weights only appear in the corresponding personalized models, which are injected at each class tag. Both the class-based LM and personalized models operate on the corresponding personalized models, which are injected at each class tag. Both the class-based LM and personalized models operate at the subword level using the algorithm described in Section 3.2.2.

### 3.3. Datasets

We evaluate on (1) a 20k utterance contact name test set and (2) a 20k utterance test set representing the general use case. Both test sets consist of anonymized data from real user interactions with a personal assistant device.

### 3.4. Second-pass rescoring

We rescore 8-best hypotheses from the first-pass shallow fusion as described in Section 3.2. Each n-best hypothesis $y_i$ can be assigned a score based on the following equation:

$$
Score(y_i) = \log P_{RNN-T}(y_i | x) + \alpha Score_{SF}(y_i) + \beta \log P_{RLM}(y_i)
$$

$P_{RLM}(y_i)$ is the probability of the hypothesis $y_i$ assigned by the rescoring LM, $Score_{SF}(y_i)$ is the shallow fusion score of $y_i$ from the first-pass. $\alpha$ and $\beta$ are the tunable scaling factors. In the tuning stage, we resort to a simulated annealing algorithm as described in [21] to find the optimal values of $\alpha$ and $\beta$. The objective of the optimization is to minimize the overall WER of the dev set. This approach enables us to optimally combine multiple rescoring LMs at the subword level using the algorithm described in Section 3.2.2.

#### 4. RESULTS AND DISCUSSION

We report on word error rate reduction (WERR) and oracle WERR to the baseline RNN-T model throughout. The oracle WERR is computed by finding the hypothesis in the 8-best that minimizes WER for each utterance.

---

**Table 1:** Results using only the contact names personalized model, with different biasing weights, and comparing word-level biasing (word), with subword-level biasing with (ctxt-subwd) and without context (noctxt-subwd)

<table>
<thead>
<tr>
<th>Model</th>
<th>Contacts WERR</th>
<th>Contacts Oracle WERR</th>
<th>General WERR</th>
<th>General Oracle WERR</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>+word(1.0)</td>
<td>-6.5</td>
<td>-2.3</td>
<td>0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>+word(1.5)</td>
<td>-6.5</td>
<td>-2.5</td>
<td>1.4</td>
<td>0.1</td>
</tr>
<tr>
<td>+word(2.0)</td>
<td>-5.0</td>
<td>-1.6</td>
<td>3.0</td>
<td>0.1</td>
</tr>
<tr>
<td>+nocxt-subwd(1.0)</td>
<td>-13.3</td>
<td>-10.9</td>
<td>-0.5</td>
<td>0.0</td>
</tr>
<tr>
<td>+nocxt-subwd(1.5)</td>
<td>-14.2</td>
<td>-14.4</td>
<td>0.1</td>
<td>0.0</td>
</tr>
<tr>
<td>+nocxt-subwd(2.0)</td>
<td>-12.7</td>
<td>-16.9</td>
<td>2.0</td>
<td>0.6</td>
</tr>
<tr>
<td>+nocxt-subwd(2.5)</td>
<td>-7.8</td>
<td>-18.4</td>
<td>5.0</td>
<td>0.8</td>
</tr>
<tr>
<td>+nocxt-subwd(3.0)</td>
<td>-0.4</td>
<td>-18.1</td>
<td>9.6</td>
<td>1.7</td>
</tr>
<tr>
<td>+ctxt-subwd(1.0)</td>
<td>-14.0</td>
<td>-10.9</td>
<td>-0.5</td>
<td>0.0</td>
</tr>
<tr>
<td>+ctxt-subwd(1.5)</td>
<td>-16.3</td>
<td>-13.6</td>
<td>-0.3</td>
<td>0.0</td>
</tr>
<tr>
<td>+ctxt-subwd(2.0)</td>
<td>-16.5</td>
<td>-16.9</td>
<td>0.6</td>
<td>0.3</td>
</tr>
<tr>
<td>+ctxt-subwd(2.5)</td>
<td>-14.3</td>
<td>-16.9</td>
<td>2.7</td>
<td>1.1</td>
</tr>
<tr>
<td>+ctxt-subwd(3.0)</td>
<td>-10.8</td>
<td>-16.7</td>
<td>5.0</td>
<td>2.5</td>
</tr>
</tbody>
</table>

---

**4.1. Comparing shallow fusion approaches**

In Table 1, we report results comparing word-level biasing, to subword-level biasing with and without context, using different biasing weights. We report results using only the personalized contact names model for shallow fusion. We find significant improvements in WERR and oracle WERR when applying biasing at the subword level (best WERR improvement: 14.2%) compared to the word level (best WERR improvement: 6.5%). This aligns with previous work [10, 13], which found that applying biasing at the subword level allows more critical words to stay on the beam.

Comparing the subword results with and without context, we observe larger improvements in WERR on contact names at higher biasing weights when using the contextual biasing model. For example, at a weight of 2.5, we observe improvements of 14.3% with context, but only 7.8% without context. Additionally, we observe that constraining shallow fusion with context decreases the impact on the general WERR at higher biasing weights.

Finally, we observe that increasing the biasing weight leads to improvements in oracle WERR on contact names, even when overall WERR improvements decrease. This suggests that a higher weight allows for more personalized content to appear in the n-best hypotheses, even as it increases the number of false recognitions in the 1-best hypothesis. We return to this point later.
We observe that de-biasing is especially useful when there is no personalized content without compromising the WER of the general test set.

4.3.2. Adding additional personalized models in shallow fusion

Second-pass optimization can not only recover from degradation in WERR but can also improve WERR when additional personalized models are added to first-pass shallow fusion without context (Table 4). The first-pass degradation can be seen in Table 2 and is reproduced in Table 4. We observe that in general, incorporating more biasing models without context results in larger degradations on the general test set. However, second-pass rescoring with de-biasing enables us to completely recover from these degradations, while continuing to improve overall contact name WERR. This aligns with our reasoning that second-pass can improve the first-pass 1-best degradation as long the first-pass oracle WERR continues to improve.

In Table 5, we report the WERR post second-pass rescoring for various weights of shallow fusion biasing, with and without context. As the biasing weight increases, we improve WERR for both the Contacts and General test sets.

5. CONCLUSION

In this work, we have presented several strategies to improve personal content recognition for end-to-end speech recognition systems. We have outlined a novel algorithm for efficient biasing of personalized content on the subword level at inference time. This helps us improve on personal content recognition by 14% – 16% compared to RNN-T. We also describe a novel second-pass optimization to improve recognition by an additional 13% - 15% without degrading the general use case. Combining the two strategies, we achieve 27% - 30% improvement overall in personal content recognition and about 2.5% improvement on the general test set. We also elucidate ways to tackle degradation on the general test set when biasing the RNN-T model in the absence of any context.
6. REFERENCES


