Attention-based Contextual Language Model Adaptation for Speech Recognition

Richard Diehl Martinez, Scott Novotney, Ivan Bulyko,
Ariya Rastrow, Andreas Stolcke, Ankur Gandhe

Amazon Alexa, Seattle, WA, USA
{mrtimri,snovotne,ibbulyko,arastrow,stolcke,aggandhe}@amazon.com

Abstract

Language modeling (LM) for automatic speech recognition (ASR) does not usually incorporate utterance level contextual information. For some domains like voice assistants, however, additional context, such as time at which an utterance was spoken, provides a rich input signal. We introduce an attention mechanism for training neural speech recognition language models on both text and non-linguistic contextual data [1]. When applied to a large de-identified dataset of utterances collected by a popular voice assistant platform, our method reduces perplexity by 7.0% relative over a standard LM that does not incorporate contextual information. When evaluated on utterances extracted from the long tail of the dataset, our method improves perplexity by 9.0% relative over a standard LM and by over 2.8% relative when compared to a state-of-the-art model for contextual LM.

1 Introduction

Conventional automatic speech recognition (ASR) systems include a language model (LM) and an acoustic model (AM). The LM component is trained separately, typically on large amounts of transcribed utterances that have been collected by an existing speech recognition system.

Voice assistants have become ubiquitous and crucially rely on ASR systems to convert user inputs to text. They often collect utterances spoken by users, along with associated de-identified contextual information. We hypothesize that these additional data, such as the time at which an utterance was spoken, provide a useful input signal for a LM. As an example, knowing that an utterance was spoken on December 25th, a LM should learn that the word "christmas" rather than "easter" is more likely to follow the phrase "lookup cookie recipes for".

[1]We make a large portion of our code available under: https://github.com/amazon-research/contextual-attention-nlm

To-date, some voice assistants have leveraged coarse geographic information for improving location search queries (Bocchieri and Caseiro, 2010; Lloyd and Kristjansson, 2012). These past efforts, however, have largely focused on improving a particular skill of an ASR system, and not the system's speech recognition in general.

In this paper, we focus on adapting recurrent neural network language models (RNN-LMs) to use both text and non-linguistic contextual data for speech recognition in general. While, outside of ASR, transformer-based (Vaswani et al., 2017) language models have largely replaced RNN-LMs, RNNs remain dominant in ASR architectures such as connectionist temporal classification (Graves et al., 2006), and RNN-T (Graves, 2012; He et al., 2019).

The most common method for incorporating non-linguistic information into a RNN-LM is to learn a representation of the context that is concatenated with word embeddings as input to the model. This concatenation-based approach has been used in a variety of domains including text classification (Yogatama et al., 2017), personalized conversational agents (Wen et al., 2013), and voice search queries (Ma et al., 2018).

Recently, attention mechanisms, initially developed for machine translation (Bahdanau et al., 2014; Luong et al., 2015), have been used by neural LMs to adaptively condition their predictions on certain non-linguistic contexts. Tang et al. (2016) use an attention module that attends to word-location information to predict the polarity of a sentence. Similarly, Zheng et al. (2019) use an attention mechanism over learned personality trait embeddings, in order to generate personalized dialogue responses.

The aforementioned approaches learn a representation of the context that is directly used as input to a neural LM. In contrast to these methods, Jaech and Ostendorf (2018a) adapt the weight-

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1994–2003

matrix used in an RNN model to a given contextual input. The authors propose decomposing the weight-matrix into a set of left and right basis tensors which are then multiplied by a learned context embedding to produce a new weight-matrix. Factorizing the weight-matrix enables a larger fraction of a model's parameters to adjust to a given contextual signal. This factorization-based approach has proven effective in generating automatic completions of sentences that are personalized for particular users (Jaech and Ostendorf, 2018b).

We introduce an attention mechanism that augments both the concatenation-based and factorization-based approaches to condition a neural LM on context. The attention mechanism that we propose builds up a dynamic context representation over the course of processing an utterance. The resulting embedding can be used as an additional input to either the concatenation-based or factorization-based model.

Our experiments focus primarily on conditioning neural LMs on datetime context. We concentrate on datetime information because of its widespread availability in many ASR systems. Our approach, however, can generalized to any type of context. To underscore this point we also provide results for conditioning LMs on geolocation information and dialogue prompts that are commonly available in ASR systems.

We evaluate our method on a large de-identified dataset of transcribed utterances. Compared to a standard model that does not include contextual information, using our method to contextualize a neural LM on datetime information achieves a relative reduction in perplexity of 7.0%, and a relative reduction in perplexity of 9.0% when evaluated on the tail of this dataset. Moreover, our attention mechanism can improve state-of-the-art methods for conditional LMs by over 2.8% relative in terms of perplexity.

2 Data

We use a corpus of over 5,000 hours of de-identified, transcribed English utterances, collected over several years. Each utterance also contains associated information about the year, month, day, and hour that the utterance was spoken. The datetime information is reported according to the local time zone of each given user. Any information about the device or the speaker from which an utterance originates has been removed. We randomly split our dataset into a training set, development set and test set, using a partition ratio of 90/5/5 and we ensure that each partition contains more than 500 hours worth of data.

3 Context Representation

A typical utterance in our dataset might look like this:

2020-12-23 07:00 play christmas music.

In the example above, we can infer that the utterance "play christmas music" was spoken on December 23, 2020 at 7 in the morning local time. In order to condition a LM on this datetime information, we consider two methods for transforming the contextual information into a continuous vector representation:

1. **Learned embeddings:** We first consider creating tokens for the month number, week number, day of the week and hour that an utterance was spoken. In the example above, we would transform the datetime information into tokens representing: month-12, week-52, wednesday, 7am. These tokens are subsequently used as input to the model, where they are passed through an embedding layer to generate context embeddings. These embeddings are initialized as random vectors, and trained along with the rest of the model. We experiment with different ways of parsing the information, such as encoding weekday versus weekend, or morning versus evening, but find this information is largely entailed within our method for processing datetime information.

2. **Feature-engineered representation:** Additionally, we consider transforming the datetime information into a single 8-dimensional feature-engineered vector, where the dimensions of the vector are defined as

$$\begin{bmatrix} sin(\frac{2\pi \cdot hour}{24}) & cos(\frac{2\pi \cdot hour}{24}) \\ sin(\frac{2\pi \cdot day}{7}) & cos(\frac{2\pi \cdot day}{7}) \\ sin(\frac{2\pi \cdot week}{53}) & cos(\frac{2\pi \cdot week}{53}) \\ sin(\frac{2\pi \cdot month}{12}) & cos(\frac{2\pi \cdot month}{12}) \end{bmatrix}$$

Since the datetime context is continuous and cyclical, this approach explicitly encodes tem-

poral proximity in the date and time information.

3.1 Input Representation

We assume as input to a model a sequence of either word or subword tokens, w_i for $i \in \{1, \ldots, n\}$, that are converted by an embedding layer into embeddings $x_i \in \mathcal{R}^e$ for $i \in \{1, \ldots, n\}$, where *n* is the length of the input sequence and *e* is the dimensionality of the word embeddings.

We additionally represent the contextual information as either:

1. A set, M, of four learned context embeddings $M = \{m_1, m_2, m_3, m_4\}$, where m_1 is an encoding of the month information, m_2 is an encoding of the week information, m_3 is an encoding of the day of the week information, and m_4 is an encoding of the hour of the day information. When using the concatenation-based or factorization-based approaches without attention, we first concatenate the embeddings together, $m = [m_1; m_2; m_3; m_4]$, and use the resulting vector as input to the model.

2. A set, M, containing a single embedding $M = \{m\}$, where m represents an 8-dimensional feature-engineered contextual datetime representation, as described in the previous section.

4 Model

In this section, we first describe the architecture of the concatenation-based and factorization-based approaches. We then introduce our attention-mechanism that can be used to augment both of these approaches. The notation we use to describe architectures assumes a 1-layer RNN model. The methods we discuss, however, can be applied to each layer of a multi-layer RNN model.

4.1 Concatenation-based LM Adaptation

The concatenation-based approach learns a weight matrix $\mathbf{W_m}$ of dimensionality $\mathcal{R}^{f \times d}$, where *f* represents the size of the context representation and *d* represents the hidden-dimensionality of the RNN model. In practice, *f* is either a hyperparameter when datetime context is represented as learned embeddings, or $f = 8$ when this context is represented as a feature-engineered vector. When representing contextual information as learned embeddings, recall that we first concatenate the embeddings together before passing these into the model. In this case, *f* is four-times the size of each individual context embedding.

A standard RNN model without contextual information keeps track of a hidden-state at time-step *t*, h_t, that is calculated as

$$h_t = \sigma(\mathbf{W_x} x_t + \mathbf{W_h} h_{t-1} + b),$$

where x_t represents the word embedding at time-step *t*, *b* is a bias vector, $\mathbf{W_x} \in \mathcal{R}^{e \times d}$, and $\mathbf{W_h} \in \mathcal{R}^{d \times d}$.

In the concatenation-based approach, this hidden-state is adapted in the following manner

$$h_t = \sigma(\mathbf{W_m} m + \mathbf{W_x} x_t + \mathbf{W_h} h_{t-1} + b).$$

Notice that the expression above can be equivalently calculated by concatenating the matrices $\mathbf{W_m}$ and $\mathbf{W_x}$, as well as the vectors m and x_t

$$h_t = \sigma([\mathbf{W_x}; \mathbf{W_m}][x_t; m] + \mathbf{W_h} h_{t-1} + b).$$

To generate a prediction, $\hat{y}_t$ for a word at time-step *t*, h_t is passed through a projection layer, $W_v \in \mathcal{R}^{d \times |V|}$ to match the dimension of the vocabulary size $|V|$, before applying a softmax layer

$$\hat{y}_t = softmax(W_v h_t).$$

4.2 Factorization-based LM Adaptation

Unlike the concatenation-based approach, which directly inserts contextual information into the RNN cell, the factorization-based method adapts the weight matrices $\mathbf{W_x}, \mathbf{W_h}$ of the RNN model. Compared to the concatenation-based architecture, this approach adapts a larger fraction of the RNN model's parameters.

The adaption process involves learning basis tensors $\mathbf{W}^{(\mathbf{L})}_{\mathbf{x}'}, \mathbf{W}^{(\mathbf{R})}_{\mathbf{x}'}$ and $\mathbf{W}^{(\mathbf{L})}_{\mathbf{h}'}, \mathbf{W}^{(\mathbf{R})}_{\mathbf{h}'}$. These basis tensors are of fixed rank *r*, where *r* is a tuned hyperparameter. The left adaptation tensors, $\mathbf{W}^{(\mathbf{L})}_{\mathbf{x}'}, \mathbf{W}^{(\mathbf{L})}_{\mathbf{h}'}$, are of dimensionality $\mathcal{R}^{f \times e \times r}$, and $\mathcal{R}^{f \times d \times r}$, respectively. The right adaptation tensors, $\mathbf{W}^{(\mathbf{R})}_{\mathbf{x}'}, \mathbf{W}^{(\mathbf{R})}_{\mathbf{h}'}$ are both of dimensionality $\mathcal{R}^{r \times d \times f}$. We can now use the contextual representation to interpolate the two sets of basis tensors to produce two new weight matrices, $\mathbf{W}'_{\mathbf{x}}$ and $\mathbf{W}'_{\mathbf{h}}$, where

$$\mathbf{W}'_{\mathbf{x}} = \mathbf{W_x} + (\mathbf{W}^{(\mathbf{L})^T}_{\mathbf{x}'} m)^T (\mathbf{W}^{(\mathbf{R})^T}_{\mathbf{x}'} m)$$

$$\mathbf{W}'_{\mathbf{h}} = \mathbf{W_h} + (\mathbf{W}^{(\mathbf{L})^T}_{\mathbf{h}'} m)^T (\mathbf{W}^{(\mathbf{R})^T}_{\mathbf{h}'} m).$$

The resulting matrices $\mathbf{W}'_{\mathbf{x}}, \mathbf{W}'_{\mathbf{h}}$ are now used as the weights in the RNN cell. A prediction, $\hat{y}_t$, is generated in the same manner as in the concatenation-based model.

4.3 Attention Mechanism

We propose an attention mechanism that augments both the concatenation-based and factorization-based approaches. We apply this mechanism to the context embeddings at each time-step of the RNN model, in order to adapt the context representation dynamically. We hypothesize that at certain time-steps within an utterance, attending to particular datetime information will facilitate the model's predictions more than other information.

For instance, assume a LM is given the phrase "what temperature will it be on friday". By the time the model has observed the words "what temperature will", we would expect the model to condition the predictions of the remaining words primarily on the hour and day information. Using an attention mechanism enables us to dynamically weight the importance that the model places on particular datetime context as the model processes an utterance.

We assume as input to the attention mechanism the same set M of context representations. However, in the case where datetime information is represented as a feature-engineered vector, we augment M to include an 8-dimensional vector of all 0s: $M = \{m, \mathbf{0}\}$. We do so because our attention mechanism builds a dynamic representation of the context by interpolating over multiple context embeddings. Thus, the attention mechanism can act as a learnable gate to limit the non-linguistic context passed into the model. We also experiment with adding a similar vector of all 0s in the case where context embeddings are learned, but find no improvement.

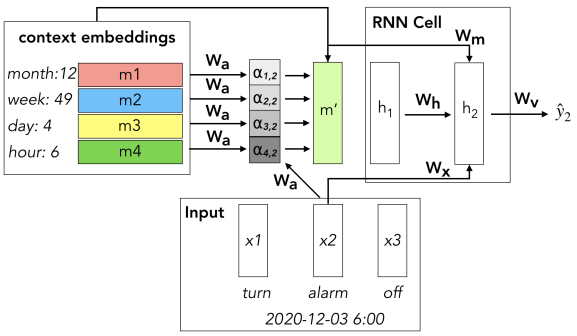

Figure 1: Model architecture of the concatenation-based model using attention. Datetime context is encoded as learned embeddings, and the input word embedding at time-step *t* is used as the query vector at *t*.

In addition to the set M, the attention mechanism takes in a query vector, q_t, for each time-step *t*. We propose two methods for defining this query vector

1. Let $q_t = x_t$, where x_t is the embedding for the input word at time-step *t*.

2. Let $q_t = h_t$, where h_t is the hidden state of the RNN model at time-step *t*.

Importantly, when q_t is chosen such that $q_t = x_t$, we can parallelize the computation of the attention mechanism for all time-steps before running a forward pass through the model. This cannot be done when $q_t = h_t$, as the attention mechanism can only be computed sequentially for each hidden state of the model.

Regardless of the choice of q_t, the attention mechanism first computes a score for each context embedding $m_i \in M$ for a given q_t. To compute this score, we learn a weight matrix W_a. The size of W_a is $\mathcal{R}^{f \times e}$ if $q_t = x_t$, or $\mathcal{R}^{f \times d}$ if $q_t = h_t$.

For a given $m_i \in M$ and q_t, we calculate a score as

$$score(m_i, q_t) = m_i^T \mathbf{W_a} q_t.$$

We then define the alignment score as

$$\alpha_{i,t} = softmax(score(m_i, q_t))$$

The alignment scores are finally used to build up a dynamic representation of the context, m'_t, for a given time-step.

$$m'_t = \sum_{i=1}^{|M|} \alpha_{i,t} m_i$$

We can now use this constructed context, as the context input to either the concatenation-based or the factorization-based approach.

In Figure 1 we illustrate how the attention mechanism augments the concatenation-based approach. For an utterance like "turn alarm off", we showcase how the model builds a dynamic representation of the datetime context, at a given time-step *t* (*t* = 2 in the figure).

5 Experimental Setup

We used a 1-recurrent-layer LSTM model (Hochreiter and Schmidhuber, 1997) as the base model in all of our experiments. Both the concatenation-based and factorization-based methods can be easily adapted to use an LSTM cell. Models were trained using the Adam optimizer with an initial learning rate of 0.001, and a standard cross entropy loss function. Each of the LMs was trained for

Method	Context Approach	Attention Query	Relative Perplexity Reduction		
			Full	Head	Tail
Default	NA	NA	0%	0%	0%
Prepend	Embeddings	NA	0.83%	**2.54%**	0.12%
Prepend	Feature-engineered	NA	**1.20%**	2.30%	**0.74%**
Concat	Embeddings	NA	6.82%	2.82%	8.68%
Concat	Embeddings	Hidden	7.00%	2.83%	8.91%
Concat	Embeddings	Word	**7.02%**	**2.89%**	**8.96%**
Concat	Feature-engineered	NA	6.82%	2.65%	8.71%
Concat	Feature-engineered	Hidden	7.0%	2.71%	8.97%
Concat	Feature-engineered	Word	6.94%	2.60%	8.94%
Factor	Embeddings	NA	3.29%	2.26%	3.93%
Factor	Embeddings	Hidden	4.82%	2.53%	5.96%
Factor	Embeddings	Word	5.40%	**2.58%**	6.71%
Factor	Feature-engineered	NA	5.44%	2.00%	**7.10%**
Factor	Feature-engineered	Hidden	**5.57%**	2.31%	6.82%
Factor	Feature-engineered	Word	5.05%	2.25%	6.31%

Table 1: Test set perplexities of contextual LMs based on datetime information, with relative reductions compared to the *Default* LSTM model that does not use contextual datetime information. We bold best results within each type of method, and underline best results overall. Improvements in perplexity from using our attention mechanism are statistically significant.

400,000 batch update steps, using a batch-size of 256. The training of each model was conducted on a single V100 GPU, with 16GB of memory on a Linux cluster, and took roughly 6 hours to train. Implementation of the model and training procedure was written in PyTorch and native PyTorch libraries. We used a fixed dimensionality of 512 for word, context and hidden state embeddings. We initially experimented with smaller and larger embedding sizes (50, 100, 1024), but found that 512 generally provided a good tradeoff between model performance and compute resources required to train a model. We set the rank of the basis tensors in the factorization-approach to 5, after experimenting with rank sizes 2, 3, 10, 15, 20. In practice, we found that the larger the rank size the less stable the training procedure became. Other hyperparameters, such as the initial learning rate, were selected via random search. We initialized random weights using Xavier-He weight initialization (He et al., 2015).

6 Results

6.1 Datetime

We evaluated our models on a heldout set of utterances that were randomly sampled from the full dataset. The utterances in our training and evaluation set were collected in the same time-range. We also defined the head and tail subsets of our development set, representing, respectively, the top 5% most frequently occurring utterances, and utterances occurring only once.

We used two metrics for our evaluations: perplexity and word error rate. Perplexity is a common statistic widely used in language modeling and speech recognition to measure how well a language model predicts a sample of text (Jelinek et al., 1977). Word error rate, on the other hand, measures the Levenshtein (minimum edit) distance between a recognized word sequence and a reference word sequence. In practice, these two statistics have been shown to be correlated by a power law relationship (Klakow and Peters, 2002).

In Table 1, we report the relative decrease in perplexity of models that leveraged datetime context compared to a baseline LSTM model that did not use any contextual information. We additionally trained a simple baseline, *Prepend*, which was comprised of a standard LSTM model that treated datetime context as input tokens that were prepended to the input texts.

In reporting our results, we distinguish between

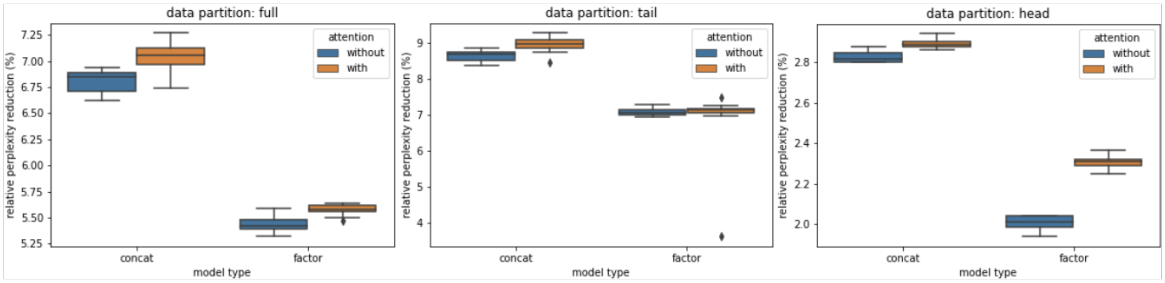

Figure 2: Perplexity confidence bounds at a 95% confidence level for the best-performing concatenation-based and factorization-based models with and without attention. Bounds are evaluated on the full, head and tail partitions of the evaluation set. Perplexity reductions are relative to the *Default* LSTM model that does not use contextual datetime information.

the two forms of representing contextual information: either as learned embeddings or as a feature-engineered representation. We also differentiate between the two variants of encoding the query vector used by the attention mechanism: either by using the hidden state vector, or the input embedding at a given time-step.

For the concatenation-based model, we found that adding our attention mechanism led to further reductions in perplexity, regardless of the type of query vector or context representation used. We obtained the best results when representing datetime information as learned embeddings and using the input embedding at a given time-step as the query vector.

We corroborated these results by computing 95% confidence intervals for the best-performing concatenation-based models with and without attention. Confidence intervals were calculated by running the training algorithm 10 times for each model type. Figure 2 visualizes the intervals.

In the case of the factorization-based approach, we achieved the lowest perplexity when the attention mechanism used the hidden state of the RNN model as the query vector and datetime information was represented as a feature-engineered vector. Again, we found that on the full dataset the improvement in perplexity by using our attention mechanism was statistically significant.

In nearly every experiment we ran, we found that our attention mechanism further reduced perplexity. The use of attention led to the largest relative improvement in the factorization-based approach when using learned context embeddings. In this instance, perplexity was reduced by 2.8% on the tail of our evaluation set, and by 2.1% on the full dataset.

Method	Context Approach	Attention Query	WERR (%)	
			Full	Tail
Default	NA	NA	0.0	0.0
Prepend	FE	NA	0.0	0.0
Concat	Emb	Word	**1.1**	**1.2**
Factor	FE	Hidden	0.8	0.8

Table 2: We report relative improvement (decrease) in WER compared to the *Default* LSTM model that does not use contextual information. Context representation approaches are feature-engineered (FE) or embeddings (Emb).

In addition to evaluating the models on relative reductions in perplexity, we also validated the downstream performance of a hybrid CTC-HMM (Graves et al., 2006) ASR system that incorporated contextual information in its LM component. As the LM component of this system, we used the best-performing models within each category of method that we report in Table 1. Table 2 summarizes the results. We evaluated the relative WER reduction(WERR) on a large test set of de-identified, transcribed utterances representative of general user interactions with Alexa, as well as on the tail of this dataset. As in Table 1, the concatenation-based model with attention mechanism achieved the largest reductions in WER.

6.2 Other Non-Linguistic Context

So far, our experiments have focused exclusively on conditioning neural LMs on datetime context. We underscore, however, that the contextual mechanism we introduce can be applied to any type of contextual information that can be represented as embeddings. To illustrate this point we train two neural LMs using two other types of context: geolocation information and dialogue prompts.

Context Type	Relative PPL Reduction (%)		WERR (%)	
	Full	Tail	Full	Tail
Default	0.0	0.0	0.0	0.0
Datetime	11.4	11.6	<u>1.6</u>	<u>1.7</u>
Geo-hash	12.4	12.5	0.5	1.0
Dialogue Prompt	<u>13.9</u>	<u>14.1</u>	0.3	0.6

Table 3: We report relative improvement (decrease) in WER and (decrease) in perplexity (PPL) compared to the *Default* LSTM model that does not use contextual information. We report results on both the full test dataset as well as utterances from the tail of the dataset. The best results are underlined.

We train the LMs on a subset of the utterances of the initial dataset which also contain utterance-level geo-hash information and dialogue prompt information. The geo-hash information [2] associated with each utterance encodes a very rough estimate of the geolocation of a user's device. Dialogue prompts indicate whether a transcribed utterance was an initial query to the dialog system or if it was a follow-up turn.

We learn embeddings to represent both the geo-hash and the dialogue prompt information. We ingest both types of contexts via the concatenation-based approach, using word-embeddings as the attention queries. We evaluate these models on a test set of de-identified utterances representative of user interactions with Alexa. Table 3 summarizes the results. In general, we find that conditioning neural LMs on each of the different types of context reduces perplexity and WER.

7 Analysis

In this section, we focus once again on datetime information to better understand how contextual LMs use datetime signal.

7.1 Datetime Context Signal

The first question we hope to answer is: to what extent can the relative improvements in perplexity and WER in the models that incorporate datetime context be explained by the additional signal from the context versus the additional parameters that these models contain?

To answer this question, we randomly shuffled the datetime information associated with each utterance in our training and test sets. For each of our best-performing models in a given category of method (prepend, concat, or factor), we retrained and evaluated those models on the dataset containing shuffled datetime information.

In Table 4, we report the relative degradation (i.e., a negative reduction) in perplexity resulting from evaluating these models on the shuffled datetime contexts. In general, if a model uses datetime information as an additional signal, we would expect the performance of the model to decrease when the datetime context is shuffled.

Method	Context Approach	Attention Query	Relative PPL Reduction (%)	
			Full	Tail
Prepend	FE	NA	-1.5	-1.2
Concat	Emb	Word	-1.6	-1.2
Factor	FE	Hidden	-1.4	-1.0

Table 4: Relative degradation in perplexity (PPL) of models that incorporate datetime information when that context is randomly shuffled. Context representations are feature-engineered (FE) or embeddings (Emb).

We observed the overall largest relative degradation in perplexity, when using the concatenation-based model. Recall that when trained on correct datetime information this was our best-performing model overall in terms of both perplexity and WER, indicating that the performance of this model can be attributed in part to its use of contextual information.

7.1.1 Visual Analysis

In addition to these results, we visualize how the contextual LMs leverage datetime contexts. For a given utterance, we can evaluate the probability of the words in the utterance as we vary the datetime information associated with the utterance. In Figure 3, we evaluate the conditional probability of the word "snooze" in an utterance following the start-of-sentence token, as we vary the hour of day information associated with this utterance. As we would expect, the probability of this "snooze" is highest in the morning (between 5 and 6 am), as users are waking up and snoozing their alarms. As we move away from the morning hours, the conditional probability of the word "snooze" decreases substantially, reaching a low-point by the afternoon and evening. The horizontal blue dashed line

[2] We use a two integer precision geo-hash.

indicates the conditional probability of the word 'snooze' following the start-of-sentence token when evaluated with a LM that does not ingest datetime information. This analysis further corroborates that the trained contextual LMs successfully condition their predictions on datetime information.

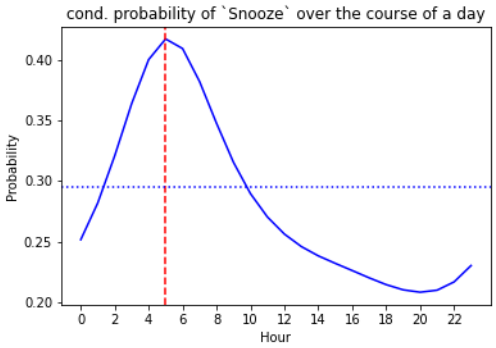

Figure 3: Changing conditional probability of the word "snooze" as the associated hour of day information varies.

7.2 Attention Weights

We next seek to understand how the attention mechanism constructs a dynamic representations of datetime context. To do so, we visualize the weights of the attention mechanism as an utterance is processed by the model. For a given utterance like "play me best christmas songs" spoken in December, we highlight the changing weight placed on each of the datetime information. Figure 4 shows this analysis.

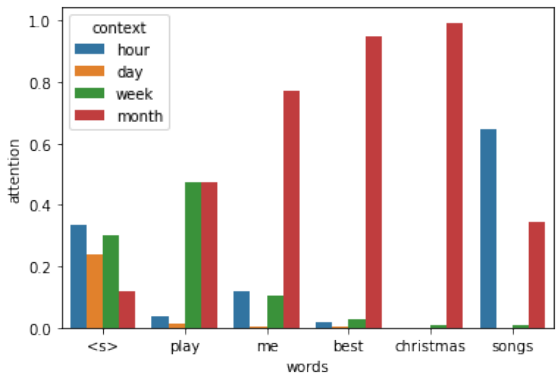

Figure 4: Changing attention weights placed on particular context embeddings over the course of an utterance.

When the model processes the start-of-sentence token, the attention mechanism weights each of the datetime information roughly equally. However as the model processes the subsequent words "play me best", the attention begins to shift towards using more of the month information (i.e., that this utterance was spoken in December), and away from hour and day information. This would suggest that conditioning on the fact that the utterance was spoken in December can help the model predict the type of media to play.

Once the model observes the word "christmas", it places all of the attention on the month information, indicating the model has successfully learned that "christmas" is a word strongly associated with a particular month (i.e., December). Finally when the word "songs" is ingested, the model substantially reduces the weight placed on month information and in turn increases the weight on hour information. This shift might indicate that the model has learned to condition the type of music users listen to to the hour of the day. Overall, the behavior of the attention mechanism is consistent with our initial hypothesis that certain types of datetime information can benefit a contextual LM model more than others over the course of an utterance.

8 Related Work

Within the domain of ASR, Biadsy et al. (2017) have explored using an adaptive-training approach to incorporate non-linguistic features into a maximum entropy LM. They propose first training the parameters of a LM that are associated with text data, then freezing those parameter and learning parameters associated with multiple types of non-linguistic features.

Zhang et al. (2019) and Yoon et al. (2017) propose a similar two-pronged approach for personalizing conversational neural LMs. They propose first pretraining a RNN-LM on a large dataset of conversational data, then finetuning the model on data associated with a particular user.

More recently, Jain et al. (2020) and Liu et al. (2020) propose attention mechanisms for conditioning RNN-T and hybrid ASR systems on words that are likely to occur in an utterance.

Another related line of research has explored learning utterance embeddings for dialogue systems using Gaussian mixture models that are enhanced with utterance-level context, such as intent (Yan et al., 2020).

Outside of ASR, our work directly builds upon the concatenation-based (Mikolov and Zweig, 2012) and factorization-based (Jaech and Ostendorf, 2018a) approaches to condition RNN-LMs on sentence context. The concatenation-based ap-

proach has been adopted as a common method for incorporating non-linguistic context into a neural LM (Yogatama et al., 2017; Wen et al., 2013; Ma et al., 2018; Ghosh et al., 2016). Methods that apply low-rank matrix factorization to RNNs are somewhat newer, and were first explored by Kuchaiev and Ginsburg (2017).

Our contribution lies first in the application of these models to ASR, and secondly their extension with an attention mechanism. The attention mechanism we propose builds on the global attention model proposed by Luong et al. (2015). While attention-based models have been used to condition neural models on particular aspects or traits (Zheng et al., 2019; Tang et al., 2016), we focus on contextual information that benefits ASR systems.

9 Conclusion

In this paper, we introduce an attention-based mechanism to condition neural LMs for ASR on non-linguistic contextual information. The proposed model dynamically builds up a representation of contextual information that can be ingested into a RNN-LM via a concatenation-based or factorization-based approach. We find that incorporating datetime context into a LM can yield a relative reduction in perplexity of 9.0% over a model that does not incorporate context. Moreover, the attention mechanism we propose can improve state-of-the-art contextual LM models by over 2.8% relative in terms of perplexity. While we focus on datetime information, we demonstrate that our approach can be applied to any type of non-linguistic context, such as geolocation and dialogue prompts.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR abs/1409.1257*.

Fadi Biadsy, Mohammadreza Ghodsi, and Diamantino Caseiro. 2017. Effectively building tera scale MaxEnt language models incorporating non-linguistic signals. In *Proc. Interspeech*.

E. Bocchieri and D. Caseiro. 2010. Use of geographical meta-data in ASR language and acoustic models. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5118–5121.

Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry Heck. 2016. Contextual LSTM (CLSTM) models for large scale NLP tasks. *arXiv preprint arXiv:1602.06291*.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 369–376.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034.

Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, et al. 2019. Streaming end-to-end speech recognition for mobile devices. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6381–6385. IEEE.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Aaron Jaech and Mari Ostendorf. 2018a. Low-rank RNN adaptation for context-aware language modeling. *Transactions of the Association for Computational Linguistics*, 6:497–510.

Aaron Jaech and Mari Ostendorf. 2018b. Personalized language model for query auto-completion. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 700–705.

Mahaveer Jain, Gil Keren, Jay Mahadeokar, and Yatharth Saraf. 2020. Contextual RNN-T for open domain ASR. In *Proc. Interspeech*.

Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63.

Dietrich Klakow and Jochen Peters. 2002. Testing the correlation of word error rate and perplexity. *Speech Communication*, 38(1-2):19–28.

Oleksii Kuchaiev and Boris Ginsburg. 2017. Factorization tricks for LSTM networks. *CoRR abs/1703.10722*.

Da-Rong Liu, Chunxi Liu, Frank Zhang, Gabriel Synnaeve, Yatharth Saraf, and Geoffrey Zweig. 2020. Contextualizing ASR lattice rescoring with hybrid pointer network language model. In *Proc. Interspeech*.

Matthew I Lloyd and Trausti Kristjansson. 2012. Acoustic model adaptation using geographic information. US Patent 8,219,384.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Min Ma, Shankar Kumar, Fadi Biadsy, Michael Nirschl, Tomas Vykruta, and Pedro Moreno. 2018. Modeling non-linguistic contextual signals in LSTM language models via domain adaptation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6094–6098. IEEE.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239. IEEE.

Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

Tsung-Hsien Wen, Aaron Heidel, Hung-Yi Lee, Yu Tsao, and Lin-Shan Lee. 2013. Recurrent neural network based personalized language modeling by social network crowdsourcing. In *Proc. Interspeech*.

Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert YS Lam. 2020. Unknown intent detection using Gaussian mixture model with an application to zero-shot intent classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1050–1060.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks. *arXiv preprint arXiv:1703.01898*.

Seunghyun Yoon, Hyeongu Yun, Yuna Kim, Gyu-tae Park, and Kyomin Jung. 2017. Efficient transfer learning schemes for personalized language modeling using recurrent neural network. *Association for the Advancement of Artificial Intelligence (AAAI) Workshop*.

Wei-Nan Zhang, Qingfu Zhu, Yifa Wang, Yanyan Zhao, and Ting Liu. 2019. Neural personalized response generation as domain adaptation. *World Wide Web*, 22(4):1427–1446.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2019. Personalized dialogue generation with diversified traits. *CoRR abs/1901.09672*.