Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 4367–4373
Marseille, 20-25 June 2022

Adapting Language Models When Training on Privacy-Transformed Data

M. A. Tuğtekin Turan[1], Dietrich Klakow[2], Emmanuel Vincent[1], and Denis Jouvet[1]
Université de Lorraine, CNRS, Inria, Loria, F-54000, Nancy, France[1]
Spoken Language Systems Group, Saarland University, Germany[2]
{tugtekin.turan,emmanuel.vincent,denis.jouvet}@inria.fr
dietrich.klakow@lsv.uni-saarland.de

Abstract

In recent years, voice-controlled personal assistants have revolutionized the interaction with smart devices and mobile applications. The collected data are then used by system providers to train language models (LMs). Each spoken message reveals personal information, hence removing private information from the input sentences is necessary. Our data sanitization process relies on recognizing and replacing named entities by other words from the same class. However, this may harm LM training because privacy-transformed data is unlikely to match the test distribution. This paper aims to fill the gap by focusing on the adaptation of LMs initially trained on privacy-transformed sentences using a small amount of original untransformed data. To do so, we combine class-based LMs, which provide an effective approach to overcome data sparsity in the context of n-gram LMs, and neural LMs, which handle longer contexts and can yield better predictions. Our experiments show that training an LM on privacy-transformed data result in a relative 11% word error rate (WER) increase compared to training on the original untransformed data, and adapting that model on a limited amount of original untransformed data leads to a relative 8% WER improvement over the model trained solely on privacy-transformed data.

Keywords: class-based language modeling, privacy-preserving learning, language model adaptation, speech-to-text

1. Introduction

Spoken dialogue systems aim to identify users' intents expressed in natural language and then to satisfy the corresponding requests. In the first step of any system, the input utterance is recognized with an automatic speech recognizer (ASR). State-of-the-art data-driven ASR models are trained on large amounts of speech data collected from the users (Tang et al., 2004). With the growing public awareness exemplified by the European Union's General Data Protection Regulation (GDPR), storing the user data raises serious privacy concerns (Nautsch et al., 2019). The user data might even contain critical information such as passwords, credit card numbers, or even health status. Therefore, spoken messages which contain sensitive information about the user characteristics should not be centralized in a single place.

This paper uses a data sanitization approach that removes personal information relating to persons, locations, and organization names. These named entities are identified using an automatic named entity recognition method (Li et al., 2020). Then they are hidden using word-by-word replacement by random named entities from the same entity class. Adelani et al. (2020) prove that this word-by-word replacement strategy provides formal privacy guarantees in terms of differential privacy.

The challenge is to ensure that, despite the sanitization, the performance of an ASR system trained on the sanitized data remains (almost) as good as that of an ASR system trained on original untransformed data. Unfortunately, hiding private information leads to less accurate ASR language models (LMs), due to the fact that the distribution of named entities in the privacy-transformed data does not match that in the original untransformed data. In this paper, we propose a method how to recover from the performance loss incurred when training LMs on sanitized data. To do, we employ a mixture of neural and word-based LMs alongside with class-based LMs, where each named entity category corresponds to one class (Brown et al., 1992).

Class-based LMs have proved their success for training on small datasets for fast LM adaptation (Samuelsson and Reichl, 1999; Naptali et al., 2012). By grouping words with similar distributional behavior into equivalent classes, class-based LMs have fewer parameters to train and can make predictions based on longer histories (Axelrod et al., 2015). This makes them particularly attractive in situations where word-based n-gram coverage is low due to a shortage of training data. Moreover, it has been found that neural and word-based contributions are complementary to each other and interpolation between these models usually leads to the best results (Gangireddy et al., 2016). In this paper, we also integrate long short-term memory (LSTM) based LMs, which have the ability to model longer temporal dependencies than n-grams and vanilla neural LMs (De Mulder et al., 2015).

LMs are ideally trained on a text corpus with a similar distribution to the target test data, however this is rarely feasible in practice. To circumvent this issue,

adaptation attempts to adjust the parameters of any LM so that it will perform well on target data. To do so, the LM is adapted using a smaller held-out dataset whose distribution matches that of the test data (McGraw et al., 2016). This typically yields a decrease in both perplexity (PPL) and word error rate (WER) (Chen et al., 2015). In this work, we focus on adapting an LM trained on sanitized data to the distribution of the original (untransformed) data, for which only a small amount of adaptation text is available. Specifically for word- and class-based LMs, we study fast marginal adaptation (FMA) by combining adapted unigrams with trigrams trained on a background corpus (Klakow, 2006). For the LSTM-based scheme, we implement a "pre-train and fine-tune" methodology as an adaptation strategy (Ma et al., 2017). Despite their simplicity, these methods provide a substantial improvement and are suitable in fast adaptation scenarios that use as few resources as possible. Applying the class-based idea, we were able to represent anonymous data better via named entities. Owing to linear interpolation over the adapted LMs, it is possible to recover some of the performance lost because of data sanitization.

The structure of the rest of the paper is as follows. In Section 2, we introduce the proposed method. In Sections 3 and 4, we describe the experiment setup and the results. We conclude in Section 5.

2. Methodology

Our methodology consists of a two-stage adaptation scheme. After getting sanitized data, the first part performs generic LM training with several models. Then, the next stage performs LM adaptation using a small amount of original (untransformed) data. Figure 1 provides a general overview of the proposed adaptation approach. The following subsections present the major components of our methodology.

2.1. Privacy-Transformation

Since LM adaptation depends critically on the quality of the background data, we first review how the background data is sanitized. Following the named entity recognition of the CoNLL's shared task (Sang and De Meulder, 2003), we consider sanitization of text data annotated with these four labels: persons (*PER*), organizations (*ORG*), locations (*LOC*), and miscellaneous named entities such as date or time (*MISC*). We do not identify demographic attributes like gender, age, and ethnicity of individuals, or any other potentially private information.

The sanitization is performed using the word-by-word, same-type transformation strategy of Adelani et al. (2020). Each identified named entity is kept unchanged with a certain probability. Otherwise, it is replaced by another random named entity of the same class (person name, location, or organization name). With this approach, even when the named entity recognizer has failed to detect a relevant word occurrence, an attacker cannot easily distinguish whether these words are the result of an actual transformation or not.

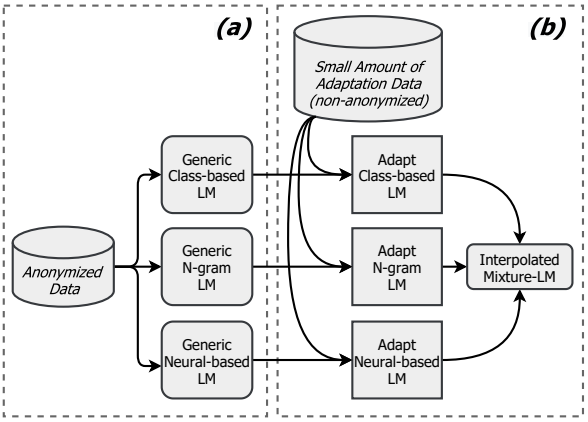

Figure 1: Block diagram of the proposed adaptation scheme. (a) corresponds to initial generic LMs using privacy-transformed text data; (b) corresponds to the adaptation to the target distribution using a limited amount of original (untransformed) text.

Individual LMs are trained on the privacy-transformed text (i.e., the sanitized data). To train class-based language models, we first find named entities in the training text data and replace them with their category tags like `LOC` or `MISC`. This yields a more realistic scenario in real-world applications. In other words, named entities are presented to the class-based LM in an unsupervised manner.

2.2. Modeling Word Classes

In this paper, we employ a class-based strategy into our fast LM adaptation framework. In the context of text sanitization described above, a sanitized group of words under a particular entity-tag can be considered close to each other compared to their inter-class relations. If we can successfully assign words to classes, it may be possible to make more reasonable predictions for histories that have private information by assuming that they are similar to other histories that we have seen.

Given the sanitized input text, the class-related probabilities can be estimated by the maximum likelihood principle which simply counts the number of occurrences of the words divided by the total number of word occurrences in that class. Moreover, conditional class probabilities can be calculated similarly. The only difference is that the word sequences used for training must be converted to class sequences.

Specifically, we propose a general way of incorporating class-based LMs with sanitized word-to-class mapping into the finite-state transducer (FST) framework. FST composition allows us to handle a class-based LM in the first decoding pass. A class-based LM can be represented by a composition of two FSTs,

namely class-map and n-gram of the class sequence. In other words, we can replace the word-based LM by cascading class-maps on word-classes instead of words.

As done by Horndasch et al. (2016), we first train an LM transducer with class entries mapped to non-terminal string identifiers like <LOC> or <PER>. Then, sub-language models for each word-class are trained. At the final stage, we insert these sub-language model WFSTs together to acquire the final LM transducer.

2.3. Adaptation of Language Models

After training the individual LMs at Stage (a) depicted in Figure 1, we apply two different adaptation strategies at Stage (b). For word- and class-based LMs, we focus on using unigram LMs to adapt trigrams. This approach combines unigram and trigram information inspired by the fast marginal adaptation (FMA) idea of Kneser et al. (1997). As an initial distribution, it uses the trigram trained on the background corpus. The desired trigram has to satisfy that its marginal is the unigram trained on the adaptation data. For a given word, w, with a history, h, the adapted LM, $P_A(w|h)$, satisfies the following constraint:

$$P_A(w|h) = \frac{1}{Z(h)} \left(\frac{P_A(w)}{P_B(w)} \right)^{\beta} P_B(w|h) \quad (1)$$

where $P_B(w)$ denotes the background LM and β is a weight parameter. An efficient way of calculating the normalization factor $Z(h)$ is provided by Kneser et al. (1997). The general idea of this adaptation scheme can be summarized as follows. The first scaling factor, $\frac{P_A(w)}{P_B(w)}$, scales up or down the probability of the word depending on whether it is more or less frequent in the adaptation data than in the background corpus. If the ratio is equal to 1, the probability remains unchanged as expected.

Eventually, we can use the unigram on the adaptation data, $P_A(w)$, as the starting distribution and then adjust it using the ratio $\frac{P_B(w|h)}{P_B(w)}$ to include history prediction. The formal derivation proceeds like generating the standard FMA using one iteration. Note that only the roles of various distributions are changed. This yields

$$P_A(w|h) = \frac{1}{Z(h)} \left(\frac{P_B(w|h)}{P_B(w)} \right)^{\beta} P_A(w). \quad (2)$$

Words w which are more likely after the particular history h are now pushed up. The efficient calculation of the normalization factor $Z(h)$ can also be applied to the updated variant of FMA with minor modifications based on the algorithm given by Kneser et al. (1997).

For the neural LSTM-LM, we adopt a fine-tuning idea where we first train a background LM on the entire sanitized training set. Then, we use this converged model to initialize the adaptation stage. The adaptation is performed by fine-tuning the final softmax layer. In other words, some layers are frozen and their weight matrices are not updated during backpropagation. Only the weights of the unfrozen layer get fine-tuned.

2.4. N-gram Approximation of Neural Language Models

This paper uses an interpolated mixture LM for final decoding. We use the Kaldi[1] speech recognition toolkit for all our experiments. The decoding can be easily done for n-gram originated word- and class-based LMs because of their ARPA-style format which is a default format inside Kaldi. For LSTM-based neural LM, it is possible to use it for lattice scoring (in a second processing pass). However, rescoring experiments, which are based on the extraction of lattices or N-best lists, depend on the n-gram models used during decoding. Therefore, we also evaluate an n-gram approximation to use LSTM-LM into the single-pass decoding to avoid introducing delay.

Adel et al. (2014) present several n-gram approximation techniques. In our experiments, we used an updated version of the probability-based conversion technique which provided better performance (Singh et al., 2017). For every word w_i of the sanitized training corpus and every associated history h corresponding to unigrams, bigrams, or trigrams, we compute the neural LM probability. Note that we do not assign count-based probabilities to n-grams; instead, we use the probabilities of our neural LM. To obtain these probabilities, we extract the LSTM-LM probability for every word w_i of the training text and assign it to the current unigrams, bigrams, and trigrams. Then, these values are averaged (if multiple occurrences exist) and normalized to obtain an approximated probability distribution. At the final stage, we smooth the distribution to provide probability mass for back-off. By doing so, we produce an n-gram ARPA LM, which is later used in the single-pass decoding.

3. Experimental Setup

We evaluate our proposed LM adaptation scheme using the Augmented Multiparty Interaction (AMI) corpus containing multi-hour meeting recordings. These meetings were recorded as part of the AMI/AMIDA projects[2] co-directed by the University of Edinburgh and Idiap. The AMI Meeting Corpus is a collection of data captured in specially instrumented meeting rooms, which recorded the multimodal signals (audio and video) for each participant. We partition the AMI data into training, adaptation, and test sets ensuring that no speaker appears in more than one set. Also, we only use speech data recorded with individual headset microphones. Table 3 presents some statistics of the data where the average utterance length is 7.5 words.

[1] Kaldi ASR: https://www.kaldi-asr.org

[2] AMI Consortium: http://www.amiproject.org

Set	Dur. (min.)	Uttr.	# of Words	
			Unique	Occurence
Train	4,880	108,221	11,882	802,604
Test	580	13,059	4,145	94,914
Adapt.	531	12,612	3,913	89,635
All	*5,991*	*133,892*	*13,079*	*987,153*

Table 1: Some statistics of the training, adaptation and test sets, from the AMI corpus. The overall duration is 100 hours, yet the unique vocabulary is limited because of its dialogue nature.

In this split, the adaptation set represents around 12% of the training data. Other splits are also investigated in the experimental evaluations to measure the impact of larger adaptation sets representing 15% and 20% of the training data. Note that the size of the test set is the same in all cases.

For the named entities, we use annotated tags given by AMI named entity instructions which mainly follow the hierarchical structure of the NIST task definition (Chinchor et al., 1999). To identify the named entities in the sanitized text, we utilized spaCy[3], which also comes with pre-trained pipelines. Eventually, we obtained 2167 unique entity tags including 226 for `LOC`, 489 for `PER`, 515 for `MISC`, and 937 for `ORG`.

During the experimental evaluation, we use word error rate (WER) and perplexity (PPL) as objective metrics. For all results presented in this paper, a matched pairs sentence-segment word error (MAPSSWE) based statistical significance test was performed at a significance level of $\alpha = 0.05$. The MAPSSWE test is essentially a parametric t-test for estimating the mean difference of normal distributions with unknown variances (Gillick and Cox, 1989). The sc_stats tool from NIST[4] was used to perform the MAPSSWE test.

In our experiments, we employ Kaldi's chain model based on a a time-delay neural network (TDNN) acoustic model (AM) architecture. The TDNN-based AM operates on 40-dimensional Mel-frequency cepstral coefficient (MFCC) features extracted from frames of 25ms length and 10ms stride, and is similar to the model specified by Peddinti et al. (2015). The speed-perturbation technique of Ko et al. (2015) is also used with a 3-fold augmentation where copies of training data are created according to factors of 0.9, 1.0, and 1.1.

[3]spaCy: `www.github.com/explosion/spaCy`
[4]NIST: `https://github.com/usnistgov/SCTK`

4. Results and Discussion

4.1. Baseline Performance

We first compare the generic LMs trained at Stage (a) of Figure 1 without any adaptation or interpolation methods. These individual LMs are the following: (M1) 3-gram word-based LM in single-pass decoding; (M2) 3-gram class-based LM in single-pass decoding; (M3) 3-gram approximation of the LSTM-based model in single-pass decoding; and finally, (M4) re-scoring of the word lattice hypotheses with the LSTM-LM (the lattices result from single-pass decoding using M1, a 3-gram word-based LM).

Table 2 shows baseline results where all these evaluations are performed over the original (untransformed) test data. The left part of this table presents the performance obtained with generic LMs trained over the sanitized input text, whereas the right part (last two columns) presents the results when using LMs trained on the original (untransformed) training data (i.e., before applying the sanitization process). For sanitized training data, the best results are obtained using either the class-based LM *M2* in first-pass decoding, or through re-scoring with the LSTM-LM in *M4*. These experiments help us understand the impact of the sanitization process. We see a large degradation between the models trained on original and sanitized data due to the privacy transformation. For example, training the 3-gram word-based model *M1* on sanitized data leads to around 11% relative WER degradation compared to training on original data. The best results are obtained when an LSTM-LM trained on original data is used in a second pass for rescoring lattice hypotheses (*M4*), but this increases the computational requirements and induces an extra delay before getting the ASR output.

Model	Sanitized Data		Original Data	
	WER [%]	PPL	WER [%]	PPL
[M1]	32.3	121	28.8	82
[M2]	**30.2**	103	29.3	74
[M3]	32.9	137	29.1	88
[M4]	30.5	103	**27.6**	73

Table 2: WER and PPL corresponding to various LMs trained on either privacy-transformed (sanitized) or original (untransformed) data.

4.2. Effect of the Adaptation Data Size

Stage (b) of Figure 1 starts with LM adaptation over untransformed adaptation data. For the results reported in Table 3, the size of the adaptation split corresponds to approximately 12% of the training split.

Using a limited amount of untransformed data, we observe the benefit of adapting LMs initially trained on a large set of sanitized data. Both WER and PPL results improves for each type of LM compared to the baseline results in Table 2. The right part of the table shows that the performance increases when the amount of untransformed adaptation data gets larger (for example 15% and 20%, instead of 12%). In all cases, the class-based LM *M2* outperforms all other LMs.

Model	Size: 12%		Size: 15%		Size: 20%	
	WER	**PPL**	**WER**	**PPL**	**WER**	**PPL**
[M1]	31.5	109	31.2	98	31.0	93
[M2]	**29.9**	94	**29.8**	91	**29.7**	86
[M3]	30.8	101	30.6	94	30.3	90
[M4]	30.1	95	29.9	91	29.8	85

Table 3: WER and PPL achieved when adapting the LMs with various amounts of (untransformed) adaptation data.

It is also interesting to evaluate the behavior when the (untransformed) adaptation data is directly included in the training data, along with the large set of sanitized data. In Table 4, the larger untransformed adaptation set yields better performance compared to Table 3.

Model	Size: 12%		Size: 15%		Size: 20%	
	WER	**PPL**	**WER**	**PPL**	**WER**	**PPL**
[M1]	31.9	116	31.4	107	29.6	90
[M2]	**30.2**	96	**29.8**	93	**29.3**	84
[M3]	31.4	106	30.5	101	29.9	92
[M4]	30.3	98	30.0	94	29.5	81

Table 4: WER and PPL achieved with various amounts of (untransformed) adaptation data used directly in LM training (in addition to the sanitized training data).

On the opposite, for the smallest size of adaptation data (12%) considered here, the performance of our proposed LM adaptation schemes are better than those achieved when the same amount of untransformed data is directly used in addition to the sanitized training set in a conventional LM training procedure.

4.3. Interpolation of the Adapted LMs

At the final stage, our methodology proposes a mixture LM for a final decoding. Note that we utilize the default adaptation split in Table 3 (corresponding to 12% of the training size) for our interpolation experiments. Thus, a linear interpolation of the previously adapted LMs is employed with the best weight combinations for 3-gram class- and word-based LMs ($\lambda_w = 0.3$ and $\lambda_c = 0.7$). Table 5 presents the results of this experiment on the first line. For both single pass decoding (using a 3-gram approximation of the LSTM-LM) and second-pass rescoring (with the LSTM-LM) schemes, we utilize $\lambda_n = 0.4$ for the LSTM-LM interpolation,

$$[(1 - \lambda_n) * (\lambda_w * P_A^w + \lambda_c * P_A^c)] + \lambda_n * P_A^n \quad (3)$$

where P_A^w, P_A^c, and P_A^n denote the adapted word-, class-, and neural LMs.

We obtain the best results with the interpolation of neural LMs at the final stage. However, it should be noted that combining word- and class-based LMs in the first place achieves only a modest improvment. We believe that in larger and more challenging datasets, the contribution of neural LMs will be greater.

Model	Description	WER	PPL
[M1 + M2]	word-& class-based	29.7	95
[M1 + M2 + M3]	+LSTM (3g app.)	29.6	92
[M1 + M2 + M4]	+LSTM (2nd-pass)	**29.4**	86

Table 5: The WER and PPL performances obtained with the linear interpolation experiments of the adapted LMs.

5. Conclusions

This paper proposes LM adaptation schemes over the privacy-transformed text for an ASR task. The sanitized text is obtained by applying data sanitization techniques using the named entities. Our models are evaluated on the AMI dialogue corpus, where we partitioned the data by distinct training, test, and adaptation splits.

We present an LM adaptation method using a class-based formulation by modifying WFSTs over the sanitized data. The word classes are determined by a named entity recognizer in an unsupervised manner and linearly interpolated with the word and neural LMs together to achieve the best results. We also investigate the adaptation of our LMs over a small amount of untransformed adaptation data. Our results prove that the adaptation is still effective with small amounts of adaptation data. Although increasing the amount of adaptation data leads to better performance in terms of PPL and WER scores, it may not feasible to obtain a large adaptation set in practical implementation.

When some relevant data is available as a-priori, it is possible to apply supervised adaptation. In our model, this style of adaptation has been shown to be successful when applied to the language model. However, if no prior in-domain text data is

available, unsupervised adaptation may not be possible under our formulation. In this paper, the proposed strategy also shows that, when only small amount of adaptation (non transformed) data is available, the adaptation is more effective than introducing the same amount of data in the training set directly during the initial training. Eventually, we show that, by hiding task-dependent named entities, we can preserve the privacy of the speakers, and still achieve comparable ASR performance with the ones before the privacy-transformation.

As a future direction, the same ideas are also extendable to a more challenging text with a large set of entity modeling. One can evaluate the proposed LM adaptation strategy to hide other private information (e.g. gender), or other topics that span diverse domains such as finance, healthcare, and politics.

6. Acknowledgements

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under the grant number 825081 (COMPRISE)[5]. Experiments presented in this paper were carried out using the Grid'5000 test-bed, https://www.grid5000.fr, supported by a scientific interest group hosted by Inria and including CNRS, RENATER, and several Universities as well as other organizations.

7. References

Adel, H., Kirchhoff, K., Vu, N. T., Telaar, D., and Schultz, T. (2014). Comparing approaches to convert recurrent neural networks into backoff language models for efficient decoding. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Adelani, D. I., Davody, A., Kleinbauer, T., and Klakow, D. (2020). Privacy guarantees for de-identifying text transformations. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Axelrod, A., Vyas, Y., Martindale, M., and Carpuat, M. (2015). Class-based n-gram language difference models for data selection. In *International Workshop on Spoken Language Translation*.

Brown, P. F., Della Pietra, V. J., Desouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480.

Chen, X., Tan, T., Liu, X., Lanchantin, P., Wan, M., Gales, M. J., and Woodland, P. C. (2015). Recurrent neural network language model adaptation for multi-genre broadcast speech recognition. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

[5] https://www.compriseh2020.eu

Chinchor, N., Brown, E., Ferro, L., and Robinson, P. (1999). Named entity recognition task definition. *Mitre and SAIC*.

De Mulder, W., Bethard, S., and Moens, M.-F. (2015). A survey on the application of recurrent neural networks to statistical language modeling. *Computer Speech & Language*, 30(1):61–98.

Gangireddy, S. R., Swietojanski, P., Bell, P., and Renals, S. (2016). Unsupervised adaptation of recurrent neural network language models. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Gillick, L. and Cox, S. J. (1989). Some statistical issues in the comparison of speech recognition algorithms. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Horndasch, A., Kaufhold, C., and Nöth, E. (2016). How to add word classes to the kaldi speech recognition toolkit. In *International Conference on Text, Speech, and Dialogue*. Springer.

Klakow, D. (2006). Language model adaptation for tiny adaptation corpora. In *International Conference on Spoken Language Processing*.

Kneser, R., Peters, J., and Klakow, D. (1997). Language model adaptation using dynamic marginals. In *European Conference on Speech Communication and Technology*.

Ko, T., Peddinti, V., Povey, D., and Khudanpur, S. (2015). Audio augmentation for speech recognition. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Li, J., Sun, A., Han, J., and Li, C. (2020). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*.

Ma, M., Nirschl, M., Biadsy, F., and Kumar, S. (2017). Approaches for neural-network language model adaptation. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

McGraw, I., Prabhavalkar, R., Alvarez, R., Arenas, M. G., Rao, K., Rybach, D., Alsharif, O., Sak, H., Gruenstein, A., Beaufays, F., et al. (2016). Personalized speech recognition on mobile devices. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Naptali, W., Tsuchiya, M., and Nakagawa, S. (2012). Topic-dependent class-based n-gram language model. *IEEE Transactions on Audio, Speech and Language Processing*, 20(5):1513–1525.

Nautsch, A., Jasserand, C., Kindt, E., Todisco, M., Trancoso, I., and Evans, N. (2019). The GDPR & speech data: Reflections of legal and technology communities, first steps towards a common understanding. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Peddinti, V., Povey, D., and Khudanpur, S. (2015).

A time delay neural network architecture for efficient modeling of long temporal contexts. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Samuelsson, C. and Reichl, W. (1999). A class-based language model for large-vocabulary speech recognition extracted from part-of-speech statistics. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Sang, E. T. K. and De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of Natural Language Learning at HLT-NAACL.*

Singh, M., Oualil, Y., and Klakow, D. (2017). Approximated and domain-adapted lstm language models for first-pass decoding in speech recognition. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Tang, M., Hakkani-Tür, D., and Tür, G. (2004). Preserving privacy in spoken language databases. In *International Workshop on Privacy and Security Issues in Data Mining*.