

G2G: TTS-DRIVEN PRONUNCIATION LEARNING FOR GRAPHEMIC HYBRID ASR

Duc Le, Thilo Koehler, Christian Fuegen, Michael L. Seltzer

Facebook AI

{duchoangle,tkoehler,fuegen,mikeseltzer}@fb.com

ABSTRACT

Grapheme-based acoustic modeling has recently been shown to outperform phoneme-based approaches in both hybrid and end-to-end automatic speech recognition (ASR), even on non-phonemic languages like English. However, graphemic ASR still has problems with low-frequency words that do not follow the standard spelling conventions seen in training, such as entity names. In this work, we present a novel method to train a statistical grapheme-to-grapheme (G2G) model on text-to-speech data that can rewrite an arbitrary character sequence into more phonetically consistent forms. We show that using G2G to provide alternative pronunciations during decoding reduces Word Error Rate by 3% to 11% relative over a strong graphemic baseline and bridges the gap on rare name recognition with an equivalent phonetic setup. Unlike many previously proposed methods, our method does not require any change to the acoustic model training procedure. This work reaffirms the efficacy of grapheme-based modeling and shows that specialized linguistic knowledge, when available, can be leveraged to improve graphemic ASR.

Index Terms— graphemic pronunciation learning, hybrid speech recognition, *chenones*, acoustic modeling

1. INTRODUCTION

There is a growing trend in the automatic speech recognition (ASR) community to use graphemes directly as the output units for acoustic modeling instead of phonemes [1–14]. Grapheme-based modeling, which does not rely on any specialized linguistic information, has been shown to outperform phoneme-based modeling in both end-to-end [5, 12] and traditional hybrid ASR [1, 13], even on non-phonemic languages with poor grapheme-phoneme relationship like English [13]. Despite achieving better overall recognition accuracy, grapheme-based modeling still has problems with rare words that are pronounced differently than how they are spelled or do not conform to the standard spelling conventions, most notably proper nouns. These problems are typically addressed in phoneme-based approaches by having linguists manually correcting the pronunciations and training a grapheme-to-phoneme (G2P) model to generalize to previously unseen words. Thanks to this extra linguistic informa-

tion, phonetic ASR may perform better than grapheme-based approaches in long-tail name recognition. It is therefore appealing to leverage linguistic knowledge in the same way to improve graphemic ASR’s performance on rare entity names.

In this work, we propose a novel method to distill linguistic knowledge into graphemic ASR by automatically learning pronunciations at the grapheme level on artificial audio generated with text-to-speech (TTS). The outcome of this method is a statistical grapheme-to-grapheme (G2G) model that can transform a character sequence into homophones with more conventional spelling, such as rewriting “Kaity” to “Katie.” We show that using G2G to generate alternative pronunciations during decoding results in **3%** to **11%** relative Word Error Rate (WER) improvement for graphemic models on utterances containing entity names. With G2G, our graphemic ASR system is able to bridge the gap with equivalent phonetic baselines on rare name recognition while achieving better overall WER. Our work addresses a long-standing weakness of grapheme-based models and contributes a novel way to combine the strengths of graphemic and phonetic ASR.

2. RELATED WORK

Grapheme-based modeling with word pieces has become the standard approach for end-to-end ASR, outperforming both context-independent phonemes and graphemes (e.g., [5–7, 11, 12]). More recently, we showed that context- and position-dependent graphemes (i.e., *chenones*) are also extremely effective for hybrid ASR, significantly outperforming *senones* [13] and achieving state-of-the-art results on Librispeech [14]. In this work, we continue to improve graphemic hybrid ASR performance on name recognition.

Existing solutions for handling proper nouns and rare words for graphemic ASR have mostly been done in the context of end-to-end systems and typically involve a TTS component to generate additional synthetic data for AM training [11, 15–17]. While the resulting improvement is promising, this approach requires careful balancing between real and synthetic data to prevent overfitting to TTS, large amount of synthesized data (relative to real data), as well as a highly diversified artificial speaker pool, all of which significantly complicate the AM training process. A different class of approach involves training a neural spelling correction

model on TTS data to fix errors made by Listen, Attend, and Spell (LAS) ASR models [18]. While this technique does not complicate AM training, it does not explicitly address the proper noun recognition problem and the WER gain they achieved on Librispeech was limited. Other methods that do not rely on TTS include leveraging phonetic information to build better word piece inventories [19] and fuzzing the training data with phonetically similar words [16, 20].

Our proposed approach differs from previous work in three ways. First, it also leverages TTS but instead produces alternative pronunciations that are ingested on-the-fly during decoding, thus does not require any change to AM training and preserves the system’s simplicity. Second, our method is geared toward conventional hybrid ASR whereas previous work mostly focused on end-to-end. Third, our work directly resolves the mismatch between graphemes and acoustic models, the fundamental cause of name recognition errors which previous work only addressed indirectly.

3. DATA

3.1. Audio Data

Our training data contains a mixture of two in-house hand-transcribed anonymized datasets with no personally identifiable information (PII). The first dataset consists of 15.7M utterances (12.5K hours) in the voice assistant domain recorded via mobile devices by 20K crowd-sourced workers. Each utterance is distorted twice using simulated reverberation and randomly sampled additive background noise extracted from public Facebook videos. The second dataset comprises 1.2M voice commands (1K hours) sampled from the production traffic of Facebook Portal¹ after the “hey Portal” wakeword is triggered. To further de-identify the user, utterances from this dataset are morphed when researchers access them (the audio is not morphed during training). We use speed perturbation [21] to create two additional copies of each utterance at 0.9 and 1.1 times the original speed. Finally, we distort each copy with additive noise sampled from the same source described previously, resulting in six copies in total ($\{0.9, 1.0, 1.1 \text{ speed}\} \times \{\text{no noise, with noise}\}$). The total amount of data after distortion is 38.6M utterances (31K hours).

Our evaluation data consists of 54.7K hand-transcribed anonymized utterances from volunteer participants in Portal’s in-house dogfooding program, which consists of employee households that have agreed to have their Portal voice activity reviewed and tested. Every utterance in this set has an associated contact list, which we use for on-the-fly personalization for calling queries. We further split this evaluation set into three subsets. Firstly, `name-prod` comprises 11.4K utterances with 3.9K unique entity names from the per-

¹Portal is a video-enabled smart device that supports calling (e.g., “hey Portal, call Alex”) and other types of voice queries (e.g., “hey Portal, what’s the weather in Menlo Park?”).

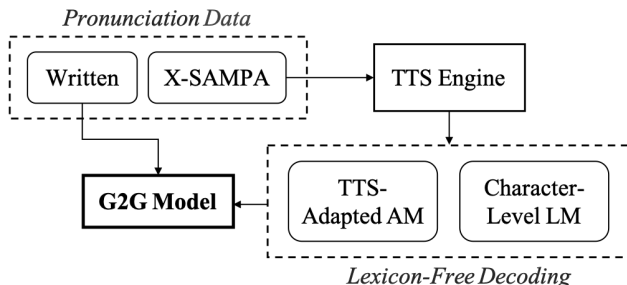


Fig. 1. Overview of TTS-based G2G training pipeline.

sonalized contact list. Secondly, `name-rare` is similar to `name-prod`, but with 800 utterances and 700 unique entity names; it contains more diverse names than those typically observed in traffic and is designed to stress-test our ASR system on name recognition. Lastly, `non-name` consists of 42.5K utterances (10.6K unique transcripts) that do not have any entity name from the associated contact list. Note that utterances from this subset may still contain other types of entities not included in the personalization data, such as city names, song names, and artist names.

3.2. Pronunciation Data

Our phonetic ASR system makes use of an in-house pronunciation corpus consisting of 1.1M unique written–pronunciation pairs, 75% of which are name pronunciations, where the written form of a word is mapped to an X-SAMPA pronunciation [22]. This dataset is used for G2P training and gives considerable advantage to phonetic ASR baselines in terms of name recognition since it has much wider name coverage than those included in the acoustic training data. The aim of this work is to distill the phonetic information from this pronunciation corpus into our graphemic ASR system.

4. G2G TRAINING METHODS

In standard grapheme-based hybrid ASR, the G2P-equivalent operation is trivial; the pronunciation of a word (e.g., “blue”) is simply its decomposed grapheme sequence (e.g., “b l u e”). However, this method may fail for long-tail words whose graphemes do not accurately reflect how they are pronounced. Having a G2P-like model for graphemes (i.e., *G2G*), is therefore desirable to handle such cases. We hereby propose two training methods for G2G based on TTS and homophones.

G2G-TTS: Figure 1 summarizes our proposed approach to train a TTS-based G2G model, which consists of three main steps. First, we feed the X-SAMPA string for each entry in our pronunciation corpus through a TTS engine to obtain the corresponding vocalization. Note that we intentionally operate on X-SAMPA directly instead of the written form to bypass the reliance on TTS’s internal G2P. Second,

we do lexicon-free decoding (LFD) [23] on these artificial utterances using a specialized character-level ASR system to generate grapheme sequences that accurately reflect their pronunciations while adhering to the English spelling convention. We will provide more detail about LFD in Section 5.2. Finally, we train a statistical G2G model to map the original written form of the pronunciation data (e.g., “Kaity”) to its LFD output (e.g., “Katie”). This is in essence a sequence-to-sequence modeling problem with many viable approaches; however, we will employ the same method as G2P model training to ensure fair comparison with phonetic baselines.

G2G-HOM: the reliance on TTS is a limitation of the previous approach. One way to circumvent TTS is to gather clusters of homophones from our phonetic lexicon (e.g., *Michael*, *Mikall*, *Mykol*) and train a model to map each word in the cluster to the cluster “root” (e.g., *Michael*). A cluster “root” is defined to be the word that most closely adheres to the English spelling convention; the exact metric for this determination will be discussed in Section 5.2. Note that this shares some similarity with the “transcript fuzzing” method utilized in [16]; however, unlike their approach which aims to increase variation during training by introducing more proper nouns, our approach aims to decrease variation during decoding.

5. EXPERIMENTAL SETUP

5.1. Baseline ASR Systems

Our baselines are hybrid ASR systems with context- and position-dependent phonemes/graphemes similar to our previous work [13]. For the phonetic setup, we train a joint-sequence G2P model [24] on the pronunciation corpus described in Section 3.2, using an in-house implementation of Phonetisaurus [25]. This model is used to generate pronunciations for both the training and decoding lexicons. For the graphemic setup, we preserve all casing information and use `unidecode` to normalize Latin characters. Excluding position-dependent variants, we have 47 phones and 56 graphemes, on which we train tri-context decision trees with 7K senones and 5K chenones.

The AM training procedure is identical across both phonetic and graphemic setups. We employ a multi-layer unidirectional Long Short-Term Memory RNN (LSTM) with five hidden layers and 800 units per layer (approximately 35M parameters). The input features are globally normalized 80-dimensional log Mel-filterbank extracted with 25ms FFT windows and 10ms frame shift; each input frame is stacked with seven right neighboring frames. The first LSTM hidden layer subsamples the output by a factor of three [26] and the target labels are delayed by 10 frames (100ms). We first train the model using Cross Entropy (CE) for 15 epochs, followed by Lattice-Free Maximum Mutual Information (LF-MMI) [27] for 8 epochs. We use Adam optimizer [28], 0.1 dropout, Block-wise Model-Update Filtering (BMUF) [29],

Written	<code>interesting</code>
LM	<code>i_B n t e r e s t i n g _ E</code>
Lexicon	<code>i_WB n t e r e s t i n g _WB</code>

Table 1. Example character-level language model and lexicon entries in our lexicon-free decoding setup.

and 64 GPUs for all AM training experiments.

Evaluation is done using our in-house one-pass dynamic decoder with n-gram language model (LM). We use a 4-gram class-based background LM with 135K vocabulary and 23M n-grams. During decoding, the `@name` class tag is expanded on-the-fly using the personalized contact list to allow for contextual biasing, and name pronunciations are obtained the same way as was done during training. For phonetic baselines, the number of pronunciation variants per name generated via G2P is a tunable hyperparameter (defaults to 2). For graphemic baselines, we add the lower-cased form of the name as an alternative pronunciation in addition to its original written form, producing up to two variants.

5.2. G2G Training and Integration

We follow the G2G training procedure outlined in Section 4.

G2G-TTS: our TTS engine is an in-house single-voice system made up of a text-processing frontend, prosody models, acoustic models, and a neural vocoder, capable of producing high fidelity speech. For LFD, we employ a 10-gram character-level LM for decoding. This LM is trained on single words obtained from the training transcripts, which captures the spelling convention of English. To preserve position dependency, we augment the characters with additional position tags: `B` (beginning of word), `E` (end of word), and `S` (singleton). These position tags allow us to map each grapheme to the correct AM output unit; for example, the lexicon entry for `i_B`, `i_E`, and `i_S` is `i_WB` (`WB` stands for “word boundary”), whereas the lexicon entry for `i` is `i` (see Table 1 for an example). The AM used for LFD is a Latency-Controlled Bidirectional LSTM (LC-BLSTM) [30] with five hidden layers and 800 units per direction per layer. This AM uses the same decision tree as our graphemic baseline; it is first trained on regular training data, then adapted with LF-MMI on TTS audio generated from the training transcripts. After adaptation, the Character Error Rate (CER) on a held-out TTS development set reduces from 1.64% to 0.16%, which denotes almost perfect recognition accuracy. The highly accurate AM coupled with the character-level LM ensures that the LFD output both captures the acoustic properties of the TTS data and closely adheres to the conventional English spelling.

G2G-HOM: our homophone-based G2G training data are 133.5K homophone clusters with at least two members, totaling 419.6K word pairs. We designate the cluster root to be the member with the highest normalized score obtained from the 10-gram character-level LM used for LFD. We hypothesize

N	Dataset	G2P	G2G-HOM	G2G-TTS
2	name-prod	9.6	9.7 [†]	
	name-rare	9.6	10.3 [†]	
	non-name	11.9	11.5 [†]	
3	name-prod	9.6	9.6	9.5
	name-rare	9.1	10.4	9.7
	non-name	11.9	11.5	11.5
4	name-prod	9.6	9.5	9.4
	name-rare	9.0	10.2	9.3
	non-name	11.9	11.5	11.5
5	name-prod	9.7*	9.5	9.4
	name-rare	8.9*	10.0	9.2
	non-name	11.9*	11.5	11.5

*: phonetic baseline †: graphemic baseline without G2G
N: maximum number of pronunciation variants

Table 2. WER comparison between phonetic (G2P) and graphemic (G2G-HOM, G2G-TTS) ASR systems.

Input	G2G-TTS Output
Kaity	K.WB a t i e_WB
Ly	L.WB e e_WB
Coce	K.WB o c h e_WB
Sera	S.WB a r a h_WB
Qifei	C.WB h i e f e_WB
Liesl	L.WB e i s e l_WB
quake	q.WB u a k e_WB
prosciutto	p.WB r o s h u t o_WB
phoneme	p.WB h o n e e m_WB
ASCII	a.WB s k y_WB

Table 3. Example G2G-TTS output for names (top half) and regular words (bottom half).

that high LM scores mean more canonical English spelling.

We apply the same G2P training method to train a joint-sequence G2G model. The resulting G2G and G2P models are therefore directly comparable; they use the same source data, the same underlying model, and the same training recipe. The output of G2G for personalized contact lists can either be used directly for decoding or mixed with the default graphemic pronunciations.

6. RESULTS AND DISCUSSION

As shown in Table 2, the graphemic ASR baseline performs better by 3% relative on non-name and similar on name-prod compared to the phonetic baseline, but underperforms significantly on name-rare by 16% relative. This large gap on name-rare confirms the limitation of default graphemic pronunciations when dealing with unconventional

names. For example, *Kaity*, *Ly*, *Coce*, *Sera*, *Qifei*, and *Liesl* are correctly recognized in the phonetic baseline but misrecognized in the graphemic baseline as *Katie*, *Lee*, *Choquel*, *Sarah*, *Shi*, and *Lisa*, respectively.

We next analyze the impact of G2G on ASR. We notice that the stand-alone G2G is good at handling the “edge cases” while the default graphemic pronunciations are more appropriate for common names, thus we report results from combining the two². Both G2G-HOM and G2G-TTS improve over the baseline, with the latter giving the best results, yielding an improvement of 3% and 11% on name-prod and name-rare, respectively. With this setup, the gap between phonetic and graphemic ASR on name-rare closes from 16% to 3% relative, while the latter performs better on the remaining test sets. We hypothesize that G2G-HOM underperforms because its output may not be fully compatible with the AM decision tree, whereas G2G-TTS guarantees this property through LFD. Further work is required to train a high-quality G2G system without relying on TTS.

We provide examples of G2G-TTS output in Table 3 to gain more understanding of how the model works and why it helps ASR. The top half of the table shows the G2G output for the misrecognized names mentioned previously, all of which have been fixed with G2G. The model is able to rewrite these names into homophone variants with more conventional spellings. Notably, the model can tell that the “Q” in *Qifei* is pronounced as “Ch;” this is a common source of errors for Chinese names in the baseline system. G2G also gives reasonable output for regular words as seen in the bottom half of the table, including spoken abbreviations like *ASCII*. Given these examples, it will be promising to leverage G2G to improve AM training. Moreover, it will also be interesting to study the impact of G2G on end-to-end ASR systems based on word pieces or context-independent graphemes.

7. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel method for training a G2G model from TTS data that can rewrite words into homophone variants with more conventional spelling. We show that G2G output can be leveraged during decoding to significantly improve graphemic ASR’s proper noun recognition accuracy and bridge the gap with phoneme-based systems on rare names. Future work will focus on further improving G2G, possibly by using neural sequence-to-sequence models, as well as applying G2G to AM training and end-to-end ASR.

8. ACKNOWLEDGMENT

We’d like to thank linguists Antony D’Avirro and Dat Vo for their help in formulating and curating data for the homophone-based G2G training approach.

²G2G-only results are consistently worse than the baseline numbers.

9. REFERENCES

- [1] M. Gales, K. Knill, and A. Ragni, “Unicode-based graphemic systems for limited resource languages,” in *Proc. ICASSP*, 2015.
- [2] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *Proc. ICASSP*, 2016.
- [3] H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition,” in *Proc. INTERSPEECH*, 2015.
- [4] H. Soltau, H. Liao, and H. Sak, “Neural Speech Recognizer: Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition,” in *Proc. INTERSPEECH*, 2017.
- [5] T. Sainath, R. Prabhavalkar, S. Kumar, S. Lee, A. Kannan, D. Rybach, V. Schogol, P. Nguyen, B. Li, Y. Wu, et al., “No Need for a Lexicon? Evaluating the Value of the Pronunciation Lexica in End-to-End Models,” in *Proc. ICASSP*, 2018.
- [6] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, “Improved training of end-to-end attention models for speech recognition,” in *Proc. INTERSPEECH*, 2018.
- [7] C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani, “State-of-the-Art Speech Recognition with Sequence-to-Sequence Models,” in *Proc. ICASSP*, 2018.
- [8] K. Audhkhasi, B. Kingsbury, B. Ramabhadran, G. Saon, and M. Picheny, “Building Competitive Direct Acoustics-to-Word Models for English Conversational Speech Recognition,” in *Proc. ICASSP*, 2018.
- [9] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, “Advancing Acoustic-to-Word CTC Model,” in *Proc. ICASSP*, 2018.
- [10] S. Ueno, H. Inaguma, M. Mimura, and T. Kawahara, “Acoustic-to-Word Attention-Based Model Complemented with Character-Level CTC-Based Model,” in *Proc. ICASSP*, 2018.
- [11] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shanguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S. Chang, K. Rao, and A. Gruenstein, “Streaming End-to-end Speech Recognition for Mobile Devices,” in *Proc. ICASSP*, 2019.
- [12] K. Irie, R. Prabhavalkar, A. Kannan, A. Bruguier, D. Rybach, and P. Nguyen, “Model Unit Exploration for Sequence-to-Sequence Speech Recognition,” in *Proc. INTERSPEECH*, 2019.
- [13] D. Le, X. Zhang, W. Zheng, C. Fuegen, G. Zweig, and M. L. Seltzer, “From Senones to Chenones: Tied Context-Dependent Graphemes for Hybrid Speech Recognition,” in *Proc. ASRU*, 2019.
- [14] Y. Wang, A. Mohamed, D. Le, C. Liu, A. Xiao, J. Mahadeokar, H. Huang, A. Tjandra, X. Zhang, F. Zhang, C. Fuegen, G. Zweig, and M. L. Seltzer, “Transformer-based Acoustic Modeling for Hybrid Speech Recognition,” in *Proc. ICASSP*, 2020.
- [15] C. Peyser, H. Zhang, T. N. Sainath, and Z. Wu, “Improving Performance of End-to-End ASR on Numeric Sequences,” in *Proc. INTERSPEECH*, 2019.
- [16] D. Zhao, T. N. Sainath, D. Rybach, P. Rondon, D. Bhatia, B. Li, and R. Pang, “Shallow-Fusion End-to-End Contextual Biasing,” in *Proc. INTERSPEECH*, 2019.
- [17] A. Rosenberg, Y. Zhang, B. Ramabhadran, Y. Jia, P. Moreno, Y. Wu, and Z. Wu, “Speech Recognition with Augmented Synthesized Speech,” in *Proc. ASRU*, 2019.
- [18] J. Guo, T. N. Sainath, and R. J. Weiss, “A Spelling Correction Model for End-to-end Speech Recognition,” in *Proc. ICASSP*, 2019.
- [19] H. Xu, S. Ding, and S. Watanabe, “Improving End-to-end Speech Recognition with Pronunciation-assisted Sub-word Modeling,” in *Proc. ICASSP*, 2019.
- [20] U. Alon, G. Pundak, and T. N. Sainath, “Contextual Speech Recognition with Difficult Negative Training Examples,” in *Proc. ICASSP*, 2019.
- [21] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, “Audio augmentation for speech recognition,” in *Proc. INTERSPEECH*, 2015.
- [22] J. K. Wells, “Computer-coding the IPA: a proposed extension of SAMPA,” in *UCL Phonetics and Linguistics*, 1995.
- [23] T. Likhomanenko, G. Synnaeve, and R. Collobert, “Who Needs Words? Lexicon-Free Speech Recognition,” in *Proc. INTERSPEECH*, 2019.
- [24] M. Bisani and H. Ney, “Joint-sequence models for grapheme-to-phoneme conversion,” *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.
- [25] J. R. Novak, N. Minematsu, and K. Hirose, “WFST-Based Grapheme-to-Phoneme Conversion: Open Source tools for Alignment, Model-Building and Decoding,” in *Proc. FSMNLP*, 2012.
- [26] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, “Low latency acoustic modeling using temporal convolution and LSTMs,” *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 373–377, 2018.
- [27] D. Povey, V. Peddinti, D. Galvez, P. Ghahramani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, “Purely sequence-trained neural networks for ASR based on lattice-free MMI,” *Proc. INTERSPEECH*, 2016.
- [28] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in *Proc. ICLR*, 2014.
- [29] K. Chen and Q. Huo, “Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering,” in *Proc. ICASSP*, 2016.
- [30] Y. Zhang, G. Chen, D. Yu, K. Yaco, S. Khudanpur, and J. Glass, “Highway long short-term memory RNNs for distant speech recognition,” in *Proc. ICASSP*, 2016.