

# R-G2P: EVALUATING AND ENHANCING ROBUSTNESS OF GRAPHEME TO PHONEME CONVERSION BY CONTROLLED NOISE INTRODUCING AND CONTEXTUAL INFORMATION INCORPORATION

Chendong Zhao<sup>★†‡</sup>Jianzong Wang<sup>†\*</sup>Xiaoyang Qu<sup>†</sup>Haoqian Wang<sup>★</sup>Jing Xiao<sup>†</sup><sup>†</sup> Ping An Technology (Shenzhen) Co., Ltd.<sup>★</sup> The Shenzhen International Graduate School, Tsinghua University, China

## ABSTRACT

Grapheme-to-phoneme (G2P) conversion is the process of converting the written form of words to their pronunciations. It has an important role for text-to-speech (TTS) synthesis and automatic speech recognition (ASR) systems. In this paper, we aim to evaluate and enhance the robustness of G2P models. We show that neural G2P models are extremely sensitive to orthographical variations in graphemes like spelling mistakes. To solve this problem, we propose three controlled noise introducing methods to synthesize noisy training data. Moreover, we incorporate the contextual information with the baseline and propose a robust training strategy to stabilize the training process. The experimental results demonstrate that our proposed robust G2P model (r-G2P) outperforms the baseline significantly (-2.73% WER on Dict-based benchmarks and -9.09% WER on Real-world sources).

**Index Terms**— grapheme-to-phoneme, transformer, synthetic noise, adversarial perturbation, contextual information

## 1. INTRODUCTION

Grapheme-to-phoneme (G2P) conversion generates a phonetic transcription from the written form of words. It is essential to develop a phonemic lexicon for TTS and ASR systems [1–4]. For this purpose, G2P techniques are used. For instance, modern TTS systems adopt G2P models as their frontend. Thus the performance of the overall system depends on the accuracy of G2P conversion. Meanwhile, TTS systems need to pronounce real-world inputs of diverse sources, like web pages or translated texts. The noise of real-world data can exhibit in many forms, such as spelling mistakes, newly emerged words, or even changed spellings over time. Unlike human who can easily pronounce these, G2P models may fail badly on such words. For example, "occurred" is commonly misspelled as "ocured". The Transformer G2P [5] converts it to [əkjʊr:d], which should be [əkɛ:d]. "pronunciation" is another common misspelling with an extra 'o', this typo results in [prɒnʌʊnsi:ɛɪfən], which should be [prɒnʌnsi:ɛɪfən].

G2P models have shown well performances on the clean text [6, 7], but they suffer from substandard real-world inputs. Indeed, G2P conversion can be considered as a neural machine translation (NMT) task, where we need to translate source graphemes into target phonemes. While many researches have focused on the robustness of NMT [8], this is the first study that evaluates and enhances the robustness of G2P conversion in real-world noisy scenarios. One of state-of-the-art models, Transformer G2P, is adopted as our baseline. Our contributions are three-fold:

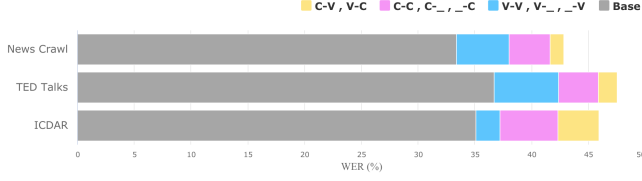
- We confirm G2P models' vulnerability to real-world noise, then statistically analyze the conversion failures. We define three noise types based on the statistics.
- We investigate three controlled noise introducing methods to synthesize our training data, which are proven to be effective in the experiments.
- We integrate and investigate the contextual information with the Transformer G2P, and propose a robust training strategy to mitigate the effect of noise during training.

## 2. REAL-WORLD NOISE EFFECT

In this Section, we study the effect of real-world noise on the baseline performance. The goals of this section are: (i) evaluating the robustness of the baseline across multimodal sources, (ii) statistically analyzing the noise distributions, and (iii) providing statistics to craft a noisy training dataset. We analyze the noise in three aspects. First, there are three major sources for G2P, including direct texts (e.g. web-crawled news), raw transcriptions from ASR, and recognitions from optical character recognition (OCR) systems. We adopt News Crawl [9], TED Talks [10] and ICDAR Post-OCR text correction [11] official dev sets as representative data which contain machine-generated noisy texts. Second, to detail into phonetic analyses, we group graphemes into **Vowels** and **Constants**. Third, there are three orthographical noise types in each group: **Insertion**, **Deletion**, and **Substitution**. Overall, we classify the reasons for G2P conversion failures (measured in word error rate) into 4 types, as shown in Fig. 1. The average increased WER caused by 3 noise types are +4.6% for vowel noise, +4.9% for

<sup>‡</sup> Work done during an internship at Ping An Technology.

<sup>\*</sup> Corresponding author: Jianzong Wang, jzwang@188.com.



**Fig. 1.** Conversion failures on multimodal sources. **Base** denotes that the text is correct but wrongly converted by the baseline. Other failures are caused by the noise, including vowel noise (in blue), constant noise (in pink) and substitutions of vowels and constants (in yellow). Take the vowel noise for example, V-V means that the failure is caused by a vowel being substituted by another vowel ("than" -> "then"). V-\_ means there's a vowel deletion noise ("neighbour" -> "neighbor") while \_-V means a vowel insertion noise ("lose" -> "loose").

constant noise, and +2.6% for substitutions of constants and vowels. Conversion failures caused by the noise account for almost 1 / 4 of all failures. This shows a great improvement space for G2P models to tackle real-world noise.

### 3. APPROACH

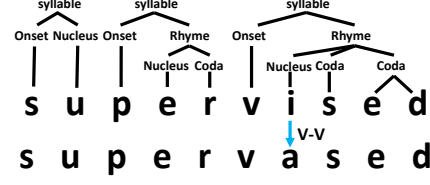
When converting a noisy word, G2P models would fail easily because the noisy grapheme pattern is not observed frequently enough in the training data. Further, other neighboring graphemes would be affected by this "less jointly trained" unexpected noise. We refer this phenomenon as "noise propagation". To address these problems, we first explore introducing carefully crafted noise in the training data, then regularize the effects of noise by the contextual information incorporation.

#### 3.1. Controlled noise introducing

Ideally, we would train a G2P model on parallel data with noisy inputs. Since no such data exists, we propose three controlled noise introducing methods. For **nat** and **syn**, each word is modified with the probability of  $p$ .

**Orthographical natural noise introduction: nat** We first explore introducing with weak noise, i.e., the natural noise. Natural noise denotes that the noisy word has similar pronunciations with the correct one but is orthographically different. We adopt the Wikipedia dataset [12] to replace correct words with their misspelled versions. It is considered weak because it may occur when one is not sure about the spelling form but aware of the pronunciation, so this type of noise wouldn't lead to big changes in the G2P conversion.

**Phonological noise synthesis: syn** Uncareful noise syntheses would easily lead to high-complexity and non-convergence for neural models. So, we synthesis noisy examples basing on the phonology knowledge and noise distributions. First, as shown in Fig. 2, graphemes could be phonologically categorized into sound units. These units can further parse into



**Fig. 2.** The illustration of a synthetic noise example.

phonetic structures, i.e., syllables. In practice, we obtain syllables using Consonant Cluster-Vowel approach [13]. Second, we limit the modifying position within the syllable boundary. This would help to constrain the "noise propagation" within the syllable boundary as well. Third, we sample and apply one of the 3 noise types in Sec. 2 to introduce with. The sampling probability is based on the collected noise distributions in Sec. 2. This 3-step noise synthesis process aims to mimic the real-world data, thus provides a realistic noisy dataset.

**Gradient-based adversarial perturbation: adv** The adversarial perturbation strategy has been applied to text classification [14] to improve robustness. Motivated by this strategy, we investigate adding the adversarial perturbation to the grapheme embedding. Let  $x, y \in \mathbb{R}^{d \times 1}$  denote the grapheme embedding and phoneme embedding, respectively.  $\hat{\theta}$  and  $\theta$  are the last and current iteration model parameters. During every training iteration, the worst case perturbation  $\delta_{ap}$  is added:

$$\delta_{ap} = \arg \min_{\|\delta\| \leq \epsilon} \log P(y | x + \delta; \theta), \quad (1)$$

where  $\epsilon$  is a hyper-parameter to control the magnitude of perturbation. Actually, it is intractable to calculate the minimum of objective function as shown in Eq. 1. In practice,  $\delta_{ap}$  is approximated via the gradient of objective function:

$$\delta_{ap} = -\epsilon g_x / \|g_x\|_2, \text{ where } g_x = \nabla_x \log P(y | x; \hat{\theta}) \quad (2)$$

where  $g_x$  denotes the gradient. The adversarial perturbation is actually a strong regularizer in the high-dimensional space, which substantially increases the diversity of inputs.

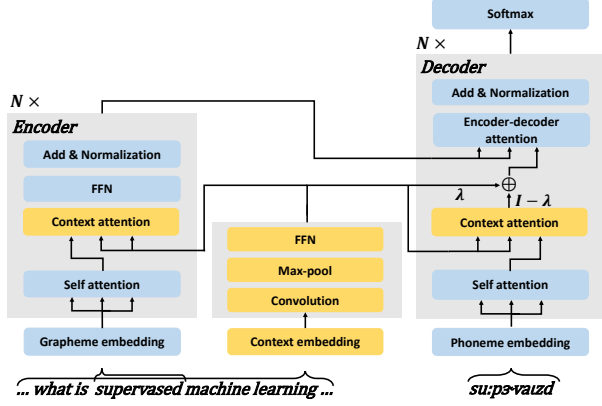
#### 3.2. Incorporating contextual information

We incorporate the contextual information to denoise and mitigate the "noise propagation" problem. To create context  $C_k$  for  $k$ -th word  $x_k$ , we simply choose  $l$  words both left and right. As shown in Fig. 3, we use a convolution layer to compute the context representation. The convolution layer aims to traverse the context embedding in turn, thus extracts feature vectors.

We integrate  $C_k$  into both the encoder and decoder of the original Transformer. In the encoder, let  $A_k$  be the output of the self-attention. The second sublayer integrates the context:

$$M_k = \text{MultiHead}(A_k, C_k, C_k). \quad (3)$$

In the decoder, let  $B_k$  be the output of the first sublayer. Similar to the encoder, the second sublayer is the context attention:



**Fig. 3.** Our extended Transformer G2P model which exploits the contextual information. The original structure is in blue, newly introduced modules are highlighted in yellow.

$$N_k = \text{MultiHead}(B_k, C_k, C_k). \quad (4)$$

We use a gating method to regulate the contextual information:

$$\text{Gating}(C_k) = \lambda C_k + (I - \lambda)N_k, \quad (5)$$

where  $I$  being a unit vector,  $\lambda$  is the gating vector given by:

$$\lambda = \text{sigmoid}(W_i C_k + W_s N_k), \quad (6)$$

where  $W_i$  and  $W_s$  are trainable parameters.  $\lambda$  represents the learned gates applied to dimensions of  $C_k$  to weight the importance. The target phoneme  $y_k$  is predicted using the softmax:

$$P(y_k | x_k, C_k; \theta) \propto \exp(W_o \times T_k), \quad (7)$$

where  $W_o \in \mathbb{R}^{|\mathcal{V}_y| \times d}$  is a model parameter,  $\mathcal{V}_y$  is the phoneme vocabulary.  $T_k \in \mathbb{R}^{d \times 1}$  is a column vector for predicting  $y_k$ . The context may provide auxiliary information (e.g., part-of-speech, word category, word frequency) for the noisy words, thus is helping to "correct" the noise in the latent space.

### 3.3. Robust training

Joint training the contextual information may increase the influence in an uncontrolled way. To avoid this, we propose a two-step training strategy that uses a clean word-level corpus  $D_w$  along with the noisy sentence-level corpus  $D_s$ . We divide model parameters into two subsets:  $\theta_w$  (highlighted in blue in Fig. 3) and  $\theta_s$  is newly-introduced (in yellow). In the first step,  $\theta_w$  are optimized on the combined corpus  $D_w \cup D_s$ :

$$\tilde{\theta}_w = \underset{\theta_w}{\operatorname{argmax}} \sum_{\langle x, y \rangle \in D_w \cup D_s} \log P(y | x; \theta_w). \quad (8)$$

In the second step,  $\theta_s$  are optimized on the  $D_s$  only:

$$\tilde{\theta}_s = \underset{\theta_s}{\operatorname{argmax}} \sum_{\langle X, Y \rangle \in D_s} \log P(y | x; \tilde{\theta}_w, \theta_s). \quad (9)$$

This is similar to the pre-training. The major difference is that ours fixes  $\tilde{\theta}_w$  when optimizing  $\theta_s$  to prevent overfitting to the noisy  $D_s$  and stabilize the training by the way.

## 4. EXPERIMENTS AND RESULTS

### 4.1. Data Preparations and Model Details

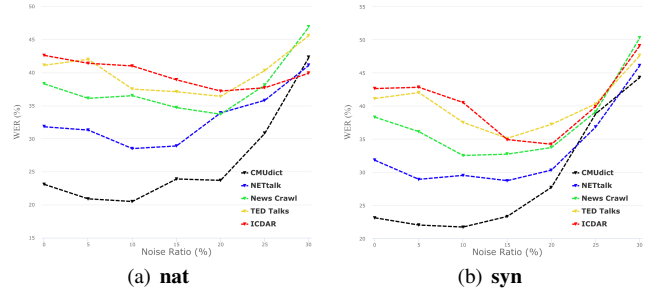
The CMUdict 0.7b [15] are adopted as  $D_w$ .  $D_s$  comes from CCOHA [16], which is clean and of large diversity in words. The test sets are two aspects: (1) **Two** dictionary-based benchmarks: CMUdict and NetTalk [17]; (2) **Three** real-world noisy sets, which are same in Sec. 2. We extract the noisy-corrected text pairs, and convert the corrected texts into ground-truth phonemes. The transformer has 4 encoder-decoder layers and 4 attention heads. The embedding size is 128 for graphemes and phonemes, 512 for word embeddings in the context. During inference, the beam search is used with size of 4.

### 4.2. Main results

Through all hyperparameters tried, we report the best results in Table 1. On the dict-based clean test sets, our models all outperform the baseline. **adv** achieves the most improvement of -2.25 % WER on CMUdict and -3.21 % WER on NetTalk. This proves our method can improve the robustness in the clean scenario. Previous methods perform well on the clean data but suffer from a great performance drop on noisy testsets. r-G2P significantly outperforms in noisy scenarios. **syn** obtains the best performance of -9.09 % WER compared with the baseline. We also compare with first correcting noisy texts by Microsoft Bing Spell Check [18], then converting by the baseline. Ours conversion accuracy is also higher. Considering the cost to integrate a spell corrector before G2P models, ours is a unified model which is more efficient and accurate.

### 4.3. Effect of noise ratio

We setup noise ratio  $p$  in the whole sentence-level corpus. Fig. 4 shows the effect of  $p$ . It is clear that introducing noise indeed enhances performance on not only the real-world noisy data but also the dict-based clean data. A moderate amount of noise would enhance the robustness, while the moderate amounts are not fixed for each dataset depending on the noise distributions. The noisy sets obviously adjust to a higher noise ratio, which is too high for clean sets. This also indicates that the different noise introducing methods have different impacts. With the same  $p$ , **syn** makes a stronger impact than **nat**.



**Fig. 4.** Varying noise ratio  $p$  results in model performances.

**Table 1.** Comparisons on various test sets. Numbers form in PER (%) / WER (%). Note that [19] has no official code.

Method	Dict-based Benchmarks		Real-world Sources		
	CMUdict	NetTalk	News Crawl	TED Talks	ICDAR
Encoder CNN, decoder Bi-LSTM (fifth model) [19]	<b>4.81</b> / 25.13	5.69 / 30.10	—	—	—
CNN with NSGD [20]	5.58 / 24.10	6.78 / 28.45	12.43 / 43.82	15.12 / 45.96	19.36 / 49.05
Encoder-decoder + global attn [21]	5.04 / 21.69	7.14 / 29.20	16.17 / 44.57	16.34 / 47.20	18.28 / 45.43
Transformer 4x4	5.23 / 22.10	6.87 / 29.82	11.10 / 42.83	15.90 / 44.56	16.01 / 42.06
Bing + Transformer 4x4	—	—	8.78 / <b>32.61</b>	10.29 / 36.94	12.15 / 37.52
r-G2P (nat)	5.22 / 20.14	6.64 / 28.85	9.94 / 33.45	<b>8.16</b> / 36.25	11.58 / 36.94
r-G2P (syn)	5.09 / 21.67	6.68 / 29.13	<b>8.61</b> / 32.76	8.64 / <b>35.06</b>	<b>10.39</b> / <b>34.35</b>
r-G2P (adv)	4.84 / <b>19.85</b>	<b>5.34</b> / <b>26.61</b>	10.31 / 36.53	9.65 / 40.72	13.46 / 39.42

#### 4.4. Effect of context length

We investigate the effect of context length and report the average WER here.  $p = 0.2$  are set as a stationary point. As shown in Table 2, incorporating the contextual information greatly enhances the anti-noise capability. Setting  $l = 1$  or  $l = 2$  as sentence-level context achieves the best performance. Using more words does not bring further improvement and increases computational cost. It's worth noticing that incorporating the contextual information has almost no contribution to **adv**. We assume that this kind of low-level feature is not helpful for perturbations in the high-dimensional hidden space. Results prove that the contextual information and robust training can cooperate with each other to improve the performance further.

**Table 2.** Context length  $l$  results in WER (%). **Ro.** denotes the robust training.  $l = 0$  means no context integrated.

	Ro.	$l = 0$	$l = 1$	$l = 2$	$l = 3$
r-G2P (nat)	—	41.29	38.74	37.08	36.61
	✓	40.37	<b>35.37</b>	35.52	35.46
r-G2P (syn)	—	40.46	39.54	36.42	36.03
	✓	39.64	36.78	<b>35.09</b>	35.28
r-G2P (adv)	—	40.15	41.23	40.98	41.39
	✓	38.23	38.51	<b>38.14</b>	38.38

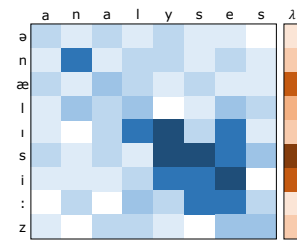
#### 4.5. Case study

Addressing out-of-vocabulary (OOV) words (i.e., abbreviations, foreign names) is a major goal for G2P models. We extract some realistic examples from Wiktionary in Table 3. Compared with the baseline, ours can generate more accurate predictions. On the other hand, CMUdict is crafted manually and updated consistently, hence prone to annotation errors. The misspelled word "commerical" is annotated with the same label as "commercial". Even for this non-existent word, ours outputs quite a convincing pronunciation.

**Table 3.** Examples converted by the baseline Transformer and ours. **GT** denotes the ground truth, where red being wrong.

	Name	Abbr.	Compound	Wrong	GT
<b>Word</b>	Xochitl	ASAP	coathanger	commerical	honest
<b>GT</b>	ʃo:tʃɪtl	eɪsæp	kəʊt hæŋɜ	kəmɜ:ʃəl	ɑ:nəst
<b>Trans</b>	zɑ:kətəl	eɪseɪpi:	kəʊθæŋɜ	kəmɜ:ʃəl	ɑ:nəst
<b>r-G2P</b>	ko:tʃɪtl	eɪsæp	kəʊt hæŋɜ	kəmɜ:ɪnkəl	ɑ:nɪst

Another issue is the homograph disambiguation, where a word pronounced differently depending on the context. For example, "analyses" is both the third-person singular form of "analyse" ( [ænəlɑɪzɪz] ) and the plural of "analysis" ( [ənælɪsɪz] ). We extract the encoder-decoder attention map and visualize it in Fig. 5. Ours took advantage of the contextual information, which mainly focused on the hard-to-distinguish phonemes, and converted this word correctly.

**Fig. 5.** Visualizations of G-P alignments and the gating  $\lambda$ . Darker color means greater alignment weights and gated scales. The context is " ... some chemical analyses of the ... ".

## 5. CONCLUSION

In this paper, we mainly raise the issue of G2P model's robustness on the noisy words since it has never been explored. We first confirm its vulnerability and statistically analyze the conversion failures caused by the noise. Then, we propose three controlled noise introducing methods to the training data. By incorporating the contextual information and the robust training process, we substantially mitigate the noise effect and achieve a robust G2P model. Experimental results show that the r-G2P significantly outperforms previous methods both on the dict-based benchmarks and in real-world scenarios.

## 6. ACKNOWLEDGMENT

This paper is supported by the Key Research and Development Program of Guangdong Province under grant No. 2021B0101400003 and the National Key Research and Development Program of China under grant No. 2018YFB0204403. Corresponding author is Jianzong Wang from Ping An Technology (Shenzhen) Co., Ltd (jzwang@188.com).

## 7. REFERENCES

- [1] Zach Ryan and Mans Hulden, “Data augmentation for transformer-based G2P,” in *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology, SIGMORPHON 2020, Online, July 10, 2020*, pp. 184–188, Association for Computational Linguistics.
- [2] Bradley Hauer, Amir Ahmad Habibi, Arnob Mallik, and Grzegorz Kondrak, “Low-resource G2P and P2G conversion with synthetic training data,” in *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology, SIGMORPHON 2020, Online, July 10, 2020*, pp. 117–122, Association for Computational Linguistics.
- [3] Ben Peters and André F. T. Martins, “One-size-fits-all multilingual models,” in *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology, SIGMORPHON 2020, Online, July 10, 2020*, pp. 63–69, Association for Computational Linguistics.
- [4] Zhenhou Hong, Jianzong Wang, Xiaoyang Qu, Jie Liu, Chendong Zhao, and Jing Xiao, “Federated Learning with Dynamic Transformer for Text to Speech,” in *Proc. Interspeech 2021*, 2021, pp. 3590–3594.
- [5] Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth, “Transformer based grapheme-to-phoneme conversion,” *Proc. Interspeech 2019*, pp. 2095–2099.
- [6] Yonghe Wang, Feilong Bao, Hui Zhang, and Guanglai Gao, “Joint alignment learning-attention based model for grapheme-to-phoneme conversion,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7788–7792.
- [7] MAC Dang-Khoa, NGUYEN, and Kim-Anh NGUYEN, “How to make text-to-speech system pronounce” volde-mort”: an experimental approach of foreign word phonemization in vietnamese,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6483–6487.
- [8] Haipeng Sun, Rui Wang, and Tiejun Zhao, “Robust unsupervised neural machine translation with adversarial denoising training,” in *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020*. 2020, pp. 4239–4250, International Committee on Computational Linguistics.
- [9] Loïc Barrault and Ondřej Bojar, “Findings of the 2019 conference on machine translation (WMT19),” in *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, Florence, Italy, Aug. 2019, pp. 1–61, Association for Computational Linguistics.
- [10] Mattia A Di Gangi, Cattoni, and Marco Turchi, “Mustc: a multilingual speech translation corpus,” in *2019 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 2012–2017.
- [11] Christophe Rigaud, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux, “Icdar 2019 competition on post-ocr text correction,” in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, pp. 1588–1593.
- [12] “Wikipedia,” [https://en.wikipedia.org/wiki/Wikipedia:Lists\\_of\\_common\\_misspellings](https://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings).
- [13] Dominic W Massaro and Michael M Cohen, “Phonological context in speech perception,” *Perception & psychophysics*, vol. 34, no. 4, pp. 338–348, 1983.
- [14] Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow, “Adversarial training methods for semi-supervised text classification,” in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.
- [15] “Cmu pronouncing dictionary,” <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>.
- [16] Reem Alatrash, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde, “Ccoha: Clean corpus of historical american english,” in *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020, pp. 6958–6966.
- [17] “NetTalk,” [https://archive.ics.uci.edu/ml/datasets/Connectionist+Bench+\(Nettalk+Corpus\)](https://archive.ics.uci.edu/ml/datasets/Connectionist+Bench+(Nettalk+Corpus)).
- [18] “Bing Spell Check,” <https://azure.microsoft.com/en-in/services/cognitive-services/spell-check>.
- [19] Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth, “Grapheme-to-phoneme conversion with convolutional neural networks,” *Applied Sciences*, vol. 9, no. 6, pp. 1143, 2019.
- [20] Moon-Jung Chae and Kyubyong Park, “Convolutional sequence to sequence model with non-sequential greedy decoding for grapheme to phoneme conversion,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2486–2490.
- [21] Shubham Toshniwal and Karen Livescu, “Jointly learning to align and convert graphemes to phonemes with neural attention models,” in *2016 IEEE Spoken Language Technology Workshop (SLT)*, 2016, pp. 76–82.