Phonological Similarity and Cognate Detection

David R. Mortensen

April 9, 2024

Phonemes can be more or less phonologically similar to one another:

(1) a. $d(\mathrm{m}, \mathrm{n}) < d(\mathrm{m}, \mathrm{t})$
b. $d(\mathrm{m}, \mathrm{t}) < d(\mathrm{m}, \mathrm{a})$
c. $d(\mathrm{i}, \mathrm{j}) < d(\mathrm{i}, \mathrm{w})$

Words can also be more or less phonologically similar to one another.

(2) a. $d(\text{tin}, \text{teen}) < d(\text{tin}, \text{tune})$
b. $d(\text{tin}, \text{dune}) < d(\text{tin}, \text{mood})$
c. $d(\text{band}, \text{pan}) < d(\text{band}, \text{banana})$

Why might we want to characterize this similarity/distance? And how could we do so?

Motivations for Phonological Similarity/Distance

Here are some tasks that motivate phonological similarity:[1]

[1] Vilém Zouhar, Kalvin Chang, Chenxuan Cui, Nathaniel Carlson, Nathaniel Robinson, Mrinmaya Sachan, and David Mortensen. Pwesuite: Phonetic word embeddings and tasks they facilitate, 2024

- **Cognate/loanword detection** [Rama, 2016, Nath et al., 2022b,a]. Along with semantic similarity, phonetic similarity measured in some latent transformation of articulatory features suggests cognacy or lexical borrowing. *This will be the basis of your final mini-project.*
- **Multilingual named entity recognition** [Bharadwaj et al., 2016, Chaudhary et al., 2018]. Phonological similarity enables cross-lingual transfer for named entity recognition since named entities will likely bear pronunciation similarities across languages.
- **Keyphrase extraction** [Ray Chowdhury et al., 2019, Fahd Saleh Alotaibi and Gupta, 2022]. Keyphrase extraction from Tweets for disaster relief can leverage phonological similarity to take advantage of the tendency for orthographic variants of the same word across different Tweets to share similar pronunciations.
- **Spelling correction** [Tan et al., 2020, Zhang et al., 2021]. Imbuing word embeddings with pronunciation similarity helps in correcting typing mistakes by substituting words with their phonetic transcription and similar-sounding words. Another approach is to pretrain a spelling-correction model on phonetic units.
- **Phonotactic learning** [Mirea and Bicknell, 2019, Romero and Salamea, 2021]. Phonetic information is a necessary part in deriving phonotactic patterns and vector representations.

- **Multimodal word embeddings** [Zhu et al., 2020, 2021]. Phonetic and syntactic information can be incorporated into semantic word embeddings.
- **Spoken language understanding** [Chen et al., 2018, 2021, Fang et al., 2020]. Training with phoneme embeddings can reduce errors from confusing phonetically similar words in automatic speech recognition so that such errors do not propagate to downstream natural language understanding tasks.
- **Language identification** [Zhan et al., 2021, Salesky et al., 2021] Phonological similarity helps in distinguishing between languages and their identification.
- **Poetry generation** [Talafha and Rekabdar, 2021, Yi et al., 2018] Word sounds and their pronunciations are critical for poetry and incorporation of this information helps in automatic poetry generation.
- **Linguistic analysis** [Hamilton et al., 2016, Ryskina et al., 2020] Apart from direct applications, there exist many investigations and analyses on what phonological and phonetic features are encoded by speakers. Phonological word similarity provides one tool by which this can be studied.

Distance Functions for Phonemes

It is possible to characterize the similarity/distance between phonemes using both a priori and empirical methods.

A Priori Distance Functions

A priori methods of estimating similarity typically involve phonological features or phonetic properties. For example, to compute the distance between two phonemes, one might take the feature vectors for the two phonemes, convert the to binary/boolean vectors, and take the Hamming distance[2]). For example, /t/ and /n/ differ in exactly two features ([voice] and [sonorant]) so the Hamming distance between them is 2.

[2] Hamming distance is the sum of the element-wise comparisons between two vectors ($a_i \neq b_i$ yields 1, $a_i = b_i$ yields 0

Empirically-Driven Distance Functions

It is also possible to learn similarity based on distribution (via phoneme embeddings). The intuition, in the cases, is that sounds that are similar occur in similar contexts[3]. These embeddings can be produced by algorithms similar to those used to learn static word embeddings.

[3] Miikka P. Silfverberg, Lingshuang Mao, and Mans Hulden. Sound analogies with phoneme embeddings. In Gaja Jarosz, Brendan O'Connor, and Joe Pater, editors, *Proceedings of the Society for Computation in Linguistics (SCiL) 2018*, pages 136–144, 2018. DOI: 10.7275/R5NZ85VD. URL `https://aclanthology.org/W18-0314`

Distance Functions for Words

Much of the time, we are interested in characterizing how far two words are from one another, not how far two sounds are from one another. This is a way of doing this based on articulatory feature vectors (using a generalization of Levensthein distance) or using a variety of embedding approaches (which are summarized in [4]).

[4] Vilém Zouhar, Kalvin Chang, Chenxuan Cui, Nathaniel Carlson, Nathaniel Robinson, Mrinmaya Sachan, and David Mortensen. Pwesuite: Phonetic word embeddings and tasks they facilitate, 2024

A Priori Distance Functions

It is possible to define a distance function, based on the algorithm for Levenshtein distance, for computing the distances between word (as matrices of feature values).

$$A_{i,j}(x,x') = \min \begin{cases} A_{i-1,j}(x,x') + d(x) \\ A_{i,j-1}(x,x') + i(x') \\ A_{i-1,j-1}(x,x') + s(x_i, x'_j) \end{cases}$$
$$A(x,x') = A_{|x|,|x'|}(x,x') \tag{1}$$

Where s is defined, roughly, as the Hamming distance between vectors:

$$s(x,x') = \frac{1}{24}\sum_{i=1}^{24} |a(x)_i - a(x')_i| \tag{2}$$

This kind of distance metric is called FEATURE DISTANCE, FEATURE EDIT DISTANCE, or ARTICULATORY DISTANCE. This metric treats all features as having the same weight, which may not be desirable.

This approach has some significant downsides: feature distance is not differentiable. It is also not terrible efficient computationally since it cannot be reduced to matrix multiplication and therefore cannot really take advantage of GPUs.

Phonetic Word Embeddings

Phonetic word embeddings[5] are dense vector representation s of words such that, if two words are phonologically similar, they will have similar embeddings. This is illustrated in Figure 1.

[5] "Phonetic word embeddings" are really phonological word embeddings, but someone used the former terminology once and it stuck.

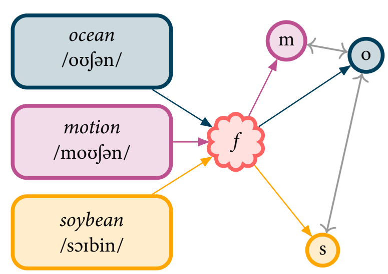

Figure 1: Embedding function *f* projects words in various forms (left) to a vector space (right) such that words with a similar pronunciation (e.g., *ocean* and *motion*) are closer than words with a dissimilar pronunciation (e.g., *ocean* and *soybean*).

Here are a variety of approaches to this problem, pasted almost verbatim from [Zouhar et al., 2024]:

Poetic Sound Similarity Parrish [2017] learns word embeddings capturing pronunciation similarity for poetry generation for words in the CMU Pronouncing Dictionary. First, each phoneme is mapped to a set of phonetic features $\mathcal{F}$ using the function $P2F : \Sigma_A \to 2^{\mathcal{F}}$. From the sequence of sets that each sequence of phonemes maps to, bi-grams of phonetic features are created (using Cartesian product $\times$ between sets a_i and a_{i+1}) and counted. The function CountVec outputs these bi-gram counts in a vector of constant dimension. The resulting vector is then reduced using PCA to the target embedding dimension d.

$$W2F(x) = \langle P2F(x_i) | x_i \in x \rangle \qquad \text{(array)} \tag{3}$$

$$F2V(a) = \text{CountVec.}\big(\bigcup_{1 \le i \le |a|-1} a_i \times a_{i+1} \big) \tag{4}$$

$$f_{PAR} = PCA_d(\{F2V(W2F(x)) | x \in \mathcal{W}\}) \tag{5}$$

The function f_{PAR} can provide embeddings even for words unseen during training. This is because the only component dependent on the training data is the PCA over the vector of bigram counts, which can also be applied to new vectors.

phoneme2vec Fang et al. [2020] do not use hand-crafted features and learn phoneme embeddings using a more complex, deep-learning, model. They start with a gold sequence of phonemes (x_i) and a noisy sequence of phonemes (y_i). The phonemes are one-hot encoded in matrices X and Y. The gold sequence is first read by an LSTM model, yielding the initial hidden state h_0. From this hidden state, the phonemes $(\hat{y}_i)$ are decoded using teacher forcing (upon predicting $\hat{y}_i$, the model receives the correct x_i as the input). The phoneme embedding matrix V is trained jointly with the model weights and constitutes the embedding function.

$$h_0 = \text{LSTM}(\text{XV}) \tag{6}$$

$$\mathcal{L}_{\text{p2v}} = -\sum_{0 < i \le |y|} \log \text{softmax}(\text{LSTM}(Y_{<i} V)_{y_i}) \tag{7}$$

For a fair comparison, we average these vectors which are *phoneme*-level to get word-level embeddings. In addition, in contrast to other embeddings, these phoneme embeddings are only 50-dimensional.

Phonetic Similarity Embeddings Sharma et al. [2021] propose a vowel-weighted phonetic similarity metric to compute similarities between words. They then use it for training phonetic word embeddings which should share some properties with this similarity function. This is in contrast to the previous approaches, where the embedding training is indirect, on an auxiliary task. Given a sound similarity function S_{PSE}, they construct a matrix of similarity scores $S \in \mathbb{R}^{|\mathcal{W}| \times |\mathcal{W}|}$ such that $S_{i,j} = S_{\text{PSE}}(\mathcal{W}_i, \mathcal{W}_j)$. On this matrix, they use non-negative matrix factorization to learn the embedding matrix $V \in \mathbb{R}^{|\mathcal{W}| \times d}$ such that the following loss is minimized:

$$\mathcal{L}_{\text{PSE}} = ||S - V \cdot V^T||^2 \tag{8}$$

Then, the i-th row of V contains the embedding for i-th word from $\mathcal{W}$. A critical disadvantage of this approach is that it cannot be used for embedding new words because the matrix V would need to be recomputed again. We apply the sound similarity function S_{PSE}, defined specifically for English, to all evaluation languages.

Count-based Vectors Perhaps the most straightforward way of creating a vector representation for a sequence of input characters or phonemes $x \in \Sigma^*$ is simply counting n-grams in this sequence. We use a term frequency-inverse document frequency (TF-IDF) vectorizer of 1-, 2-, and 3-grams (formally denoted $[x]_n$) across the input sequence of symbols (e.g. characters) with a maximum of 300 features. This vector then becomes our word embedding. For instance, the first dimension may be the TF-IDF score or occurrence count of the bigram ⟨/dɪn/, /a/⟩.

$$\mathrm{C2V}(x) = [x]_1 \cup [x]_2 \cup [x]_3 \qquad \text{(features)} \tag{9}$$

$$f_{\text{count}}(x) = \text{TF-IDF}_{\substack{\text{feat}\\\text{ures}}=d}(\{\mathrm{C2V}(x) | x \in \mathcal{W}\}) \tag{10}$$

Autoencoder Another common approach, though less interpretable, for vector representation with fixed dimension size is an encoder-decoder autoencoder. Specifically, we use this architecture together with forced-teacher decoding and use the bottleneck vector as the phonetic word embedding. In an ideal case, the fixed-size bottleneck contains all the information to reconstruct the whole sequence from Σ^*.

$$f_\theta(x) = \mathrm{LSTM}(x|\theta) \qquad \text{(encoder)} \tag{11}$$

$$d_{\theta'}(x) = \mathrm{LSTM}(x|\theta') \qquad \text{(decoder)} \tag{12}$$

$$\mathcal{L}_{\text{auto.}} = \sum_{0<i\leq|x|} -\log \mathrm{softmax}(d_{\theta'}(f_\theta(x)|x_{<i})_{x_i}) \tag{13}$$

Metric Learning As one means of generating word embeddings, we use the last hidden state of an LSTM-based model. We use characters Σ_C, IPA symbols Σ_P and articulatory feature vectors as the input.

We now have a function f that produces a vector for each input word. However, it is not yet trained to produce vectors encoding phonetic information. We, therefore, define the following differentiable loss where A is the articulatory distance.

$$\mathcal{L}_{\text{dist.}} = \frac{1}{|\mathcal{W}|} \sum_{\substack{x_a \in \mathcal{W}\\ x_b \sim \mathcal{W}}} \Big(||f_\theta(x_a) - f_\theta(x_b)||^2 - A(x_a, x_b) \Big)^2 \tag{14}$$

This forces the embeddings to be spaced in the same way as the articulatory distance (A) would space them. Metric learning (learning a function to space output vectors similarly to some other metric) has been employed previously [Yang and Jin, 2006, Bellet et al., 2015, Kaya and Bilge, 2019] and was used to train *acoustic* embeddings by Yang and Hirschberg [2019].

Triplet Margin loss While the previous approach forces the embeddings to be spaced exactly as by the articulatory distance function A, we may relax the constraint so only the structure (ordering) is preserved. This is realized by triplet margin loss:

$$\mathcal{L}_{\text{triplet}} = \max \begin{cases} 0 \\ \alpha + |f_\theta(x_a) - f_\theta(x_p)| \\ \quad - |f_\theta(x_a) - f_\theta(x_n)| \end{cases} \tag{15}$$

We consider all possible ordered triplets of distinct words (x_a, x_p, x_n) such that $A(x_a, x_p) < A(x_a, x_n)$. We refer to x_a as the anchor, x_p as the positive example, and x_n as the negative example. We then minimize $\mathcal{L}_{\text{triplet}}$ over all valid triplets. This allows us to learn θ for an embedding function f_θ that preserves the local neighbourhoods of words defined by $A(x, x')$. In addition, we modify the function f_θ by applying attention to all hidden states extracted from the last layer of the LSTM encoder. This allows our model to focus on phonemes that are potentially more useful when trying to summarize the phonetic information in a word. A related approach was used by Yang and Hirschberg [2019] to learn acoustic word embeddings. Although contrastive learning is a more intuitive approach, it yielded only negative results: $\left(\exp(|f_\theta(x_a) - f_\theta(x_p)|^2)\right) / \left(\sum \exp(|f_\theta(x_a) - f_\theta(x_n)|^2)\right)$.

Though metric learning and triplet margin loss have been applied previously to similar applications, we are the first to apply them using articulatory features and articulatory distance.

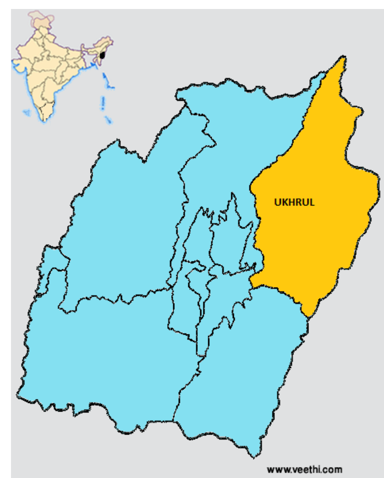

Figure 2: Map of Ukhrul District (from `https://www.veethi.com`.

Cognate Detection

COGNATES are pairs of words that are descended from the same ancestral word.

Table 1 shows some examples of cognates between Ukhrul and Huishu, two closely-related Tibeto-Burman languages of Ukhrul District, Manipur State, India.

A few things to note:

(3) a. The forms are phonologically similar
  b. There are systematic relationships between the forms. For example, word-final /a/ in Ukhrul corresponds to word-final /e/ in Huishu. Likewise, Huishu /ʔ/ corresponds to word-final /t/ or /k/ in Ukhrul.
  c. Even when the meanings of the words are not identical, they are related (e.g., ‘jump’ and ‘fly’ or ‘cry’ and ‘weep).

Two baselines for detecting cognates:

(4) a. Rank potential candidates according their string edit distance (least to greatest) from the input form

Ukhrul		Huishu	
ʃa	thick	se	thick
ka	climb	ke	climb
riŋ	alive	rɐŋ	alive
tsik	black	tsoʔ	black
tsa	eat (e.g. rice)	tse	eat
rit	heavy	rejʔ	heavy
sar	old	sa	old
kʰa	bitter	kʰe	bitter
tsat	walk	tsejʔ	walk
paj	jump	pej	fly
rak	weave	roʔ	weave
cap	cry	tsaʔ	weep

Table 1: Some Ukhrul-Huishu cognates

b. Rank potential candidates according to similarity between their bag-of-words representation of their glosses and that of the input gloss.

References

Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric learning*. Morgan & Claypool, 2015. URL `https://ieeexplore.ieee.org/abstract/document/7047350`.

Akash Bharadwaj, David R Mortensen, Chris Dyer, and Jaime G Carbonell. Phonologically aware neural model for named entity recognition in low resource transfer settings. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1462–1472, 2016. URL `https://aclanthology.org/D16-1153/`.

Aditi Chaudhary, Chunting Zhou, Lori Levin, Graham Neubig, David R Mortensen, and Jaime G Carbonell. Adapting word embeddings to new languages with morphological and phonological subword representations. *arXiv:1808.09500*, 2018. URL `https://aclanthology.org/D18-1366/`.

Qian Chen, Wen Wang, and Qinglin Zhang. Pre-training for spoken language understanding with joint textual and phonetic representation learning. In *Interspeech 2021*. ISCA, aug 2021. DOI: 10.21437/interspeech.2021-234. URL `http://dx.doi.org/10.21437/Interspeech.2021-234`.

Yi-Chen Chen, Sung-Feng Huang, Chia-Hao Shen, Hung-yi Lee, and

Lin-shan Lee. Phonetic-and-semantic embedding of spoken words with applications in spoken content retrieval. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 941–948, 2018. DOI: 10.1109/SLT.2018.8639553.

Vishal Gupta Fahd Saleh Alotaibi, Saurabh Sharma and Savita Gupta. Keyphrase extraction using enhanced word and document embedding. *IETE Journal of Research*, 0(0):1–13, 2022. DOI: 10.1080/03772063.2022.2103036.

Anjie Fang, Simone Filice, Nut Limsopatham, and Oleg Rokhlenko. Using phoneme representations to build predictive models robust to ASR errors. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 699–708. Association for Computing Machinery, 2020. ISBN 9781450380164. DOI: 10.1145/3397271.3401050. URL `https://doi.org/10.1145/3397271.3401050`.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*, 2016. URL `https://arxiv.org/abs/1605.09096`.

Mahmut Kaya and Hasan Şakir Bilge. Deep metric learning: A survey. *Symmetry*, 11:1066, 2019. URL `https://www.mdpi.com/2073-8994/11/9/1066/pdf`.

Nicole Mirea and Klinton Bicknell. Using LSTMs to assess the obligatoriness of phonological distinctive features for phonotactic learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1595–1605, jul 2019. DOI: 10.18653/v1/P19-1155. URL `https://aclanthology.org/P19-1155`.

Abhijnan Nath, Rahul Ghosh, and Nikhil Krishnaswamy. Phonetic, semantic, and articulatory features in Assamese-Bengali cognate detection. In Yves Scherrer, Tommi Jauhiainen, Nikola Ljubešić, Preslav Nakov, Jörg Tiedemann, and Marcos Zampieri, editors, *Proceedings of the Ninth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 41–53, Gyeongju, Republic of Korea, October 2022a. Association for Computational Linguistics. URL `https://aclanthology.org/2022.vardial-1.5`.

Abhijnan Nath, Sina Mahdipour Saravani, Ibrahim Khebour, Sheikh Mannan, Zihui Li, and Nikhil Krishnaswamy. A generalized method for automated multilingual loanword detection. In Nicoletta Calzolari, Chu-Ren Huang, Hansaem Kim, James Pustejovsky, Leo Wanner, Key-Sun Choi, Pum-Mo Ryu, Hsin-Hsi Chen, Lucia Donatelli, Heng Ji, Sadao Kurohashi, Patrizia Paggio, Nianwen Xue, Seokhwan Kim, Younggyun Hahm, Zhong He, Tony Kyungil Lee, Enrico Santus, Francis Bond, and Seung-Hoon Na, editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4996–5013, Gyeongju, Republic of Korea, October 2022b. International Committee on Computational Linguistics. URL https://aclanthology.org/2022.coling-1.442.

Allison Parrish. Poetic sound similarity vectors using phonetic features. In *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2017. URL https://ojs.aaai.org/index.php/AIIDE/article/view/12971.

Taraka Rama. Siamese convolutional networks for cognate identification. In *Proceedings of COLING, the 26th International Conference on Computational Linguistics*, pages 1018–1027, 2016. URL https://aclanthology.org/C16-1097/.

Jishnu Ray Chowdhury, Cornelia Caragea, and Doina Caragea. Keyphrase extraction from disaster-related tweets. In *The world wide web conference*, pages 1555–1566, 2019. URL https://dl.acm.org/doi/abs/10.1145/3308558.3313696.

David Romero and Christian Salamea. On the use of phonotactic vector representations with fasttext for language identification. *Conversational Dialogue Systems for the Next Decade*, pages 339–348, 2021. URL https://link.springer.com/chapter/10.1007/978-981-15-8395-7_25.

Maria Ryskina, Ella Rabinovich, Taylor Berg-Kirkpatrick, David R. Mortensen, and Yulia Tsvetkov. Where new words are born: Distributional semantic analysis of neologisms and their semantic neighborhoods. In *Proceedings of the Society for Computation in Linguistics*, volume 3, 2020. URL https://arxiv.org/abs/2001.07740.

Elizabeth Salesky, Badr M. Abdullah, Sabrina J. Mielke, Elena Klyachko, Oleg Serikov, Edoardo Ponti, Ritesh Kumar, Ryan Cotterell, and Ekaterina Vylomova. SIGTYP 2021 shared task: Robust spoken language identification, 2021.

Rahul Sharma, Kunal Dhawan, and Balakrishna Pailla. Phonetic word embeddings. *arXiv:2109.14796*, 2021. URL https://arxiv.org/abs/2109.14796.

Miikka P. Silfverberg, Lingshuang Mao, and Mans Hulden. Sound analogies with phoneme embeddings. In Gaja Jarosz, Brendan O'Connor, and Joe Pater, editors, *Proceedings of the Society for Computation in Linguistics (SCiL) 2018*, pages 136–144, 2018. DOI: 10.7275/R5NZ85VD. URL https://aclanthology.org/W18-0314.

Sameerah Talafha and Banafsheh Rekabdar. Poetry generation model via deep learning incorporating extended phonetic and semantic embeddings. In *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*, pages 48–55, 2021. DOI: 10.1109/ICSC50631.2021.00013.

Min Tan, Dagang Chen, Zesong Li, and Peng Wang. Spelling error correction with BERT based on character-phonetic. In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 1146–1150, 2020. DOI: 10.1109/ICCC51575.2020.9345276.

Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 2(2):4, 2006. URL http://www.cs.cmu.edu/~./liuy/frame_survey_v2.pdf.

Zixiaofan Yang and Julia Hirschberg. Linguistically-informed training of acoustic word embeddings for low-resource languages. In *Interspeech*, pages 2678–2682, 2019. URL https://www.isca-speech.org/archive_v0/Interspeech_2019/pdfs/3119.pdf.

Xiaoyuan Yi, Maosong Sun, Ruoyu Li, and Zonghan Yang. Chinese poetry generation with a working memory model, 2018.

Qingran Zhan, Xiang Xie, Chenguang Hu, and Haobo Cheng. A self-supervised model for language identification integrating phonological knowledge. *Electronics*, 10(18), 2021. ISSN 2079-9292. DOI: 10.3390/electronics10182259. URL https://www.mdpi.com/2079-9292/10/18/2259.

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. Correcting chinese spelling errors with phonetic pre-training. In *Findings of the Association for Computational Linguistics 2021*, pages 2250–2261, 2021. URL https://aclanthology.org/2021.findings-acl.198/.

Wenhao Zhu, Shuang Liu, Chaoming Liu, Xiaoya Yin, and Xiaping Xv. Learning multimodal word representations by explicitly embedding syntactic and phonetic information. *IEEE Access*, 8:223306–223315, 2020. URL https://ieeexplore.ieee.org/abstract/document/9279209/.

Wenhao Zhu, Shuang Liu, and Chaoming Liu. Incorporating syntactic and phonetic information into multimodal word embeddings using graph convolutional networks. In *ICASSP International Conference on Acoustics, Speech and Signal Processing*, pages 7588–7592. IEEE, 2021. URL https://ieeexplore.ieee.org/abstract/document/9414148/.

Vilém Zouhar, Kalvin Chang, Chenxuan Cui, Nathaniel Carlson, Nathaniel Robinson, Mrinmaya Sachan, and David Mortensen. Pwesuite: Phonetic word embeddings and tasks they facilitate, 2024.