PSET: a Phonetics-Semantics Evaluation Testbed

Gianluca Sperduti[1,2]
gianluca.sperduti@isti.cnr.it

Dong Nguyen[3]
d.p.nguyen@uu.nl

[1]**Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo" (CNR-ISTI) Pisa, Italy**
[2]University of Pisa, Pisa, Italy
[3]Utrecht University, Utrecht, The Netherlands

Abstract

We introduce the Phonetics-Semantics Evaluation Testbed (PSET), a new English-based testbed to evaluate phonetic embeddings. Our testbed is built on the assumption that phonetic embeddings should always prioritize phonetics over semantics, and it therefore leverages homophones and synonyms. We use PSET to test three phonetic embedding models: articulatory embeddings, Phoneme2Vec, and XPhoneBERT. The phonetic-based embeddings solve the task with varying degrees of success, with Phoneme2Vec performing the best. We also test five recent LLMs, GPT-4o, Gemini 2.5 Flash, Llama 3.1-8B, OLMo-7B and OLMo 2-7B. Gemini 2.5 Flash performs better than the other models. With this testbed, we hope to advance the development and evaluation of phonetic embedding models.

1 Introduction

Embeddings, numeric vector representations of text, are an important component of modern NLP systems (Incitti et al., 2023). Most embeddings are created using techniques that focus on capturing the meaning of the text and are known as *semantic embeddings*. While semantic embeddings are powerful, they are not ideal when dealing with tasks centered on sound, such as finding sound analogies (Silfverberg et al., 2018), identifying sound similarities (Parrish, 2017), detecting rhymes (Zouhar et al., 2024), detecting cognates (Batsuren et al., 2019) and improving robustness to ASR errors (Fang et al., 2020). This is where **phonetic embeddings** come into play.

Phonetic embeddings represent text based on the sound of language (Mortensen et al., 2016; Silfverberg et al., 2018; Liu et al., 2019; The Nguyen et al., 2023). Consider the following words: "hello," "yellow," and "hi,". Phonetic embeddings should represent "hello" and "yellow" close to each other in vector space due to their similar sounds, while "hi" should be more distant from these two. In contrast, with semantic embeddings, "hello" and "hi" should be close to each other due to their similar meanings, while "yellow" should be further away.

Despite the many potential uses of phonetic embeddings, there are only a few methods to evaluate them. We propose a new testbed based on the insight that phonetic embeddings should prioritize phonetic over semantic information. Suppose the embeddings primarily encode semantic information. Then, even if they also encode some phonetic information, their effectiveness for phonetic tasks (e.g., rhyme detection) would be hindered. However, existing testbeds do not focus on this issue (Zouhar et al., 2024; Efrat et al., 2023; Kolachina and Magyar, 2019). To address this gap, we propose the **Phonetics-Semantics Evaluation Testbed (PSET)**, a simple and fast method to test the quality of phonetic embeddings. It complements existing testbeds by assessing not just the presence but the predominance of phonetic information over semantic information in the model's encoding. PSET can be used with any method that can assess similarities between words, including static embedding models and prompting LLMs.

Contributions. With **PSET**, we (a) provide a new testbed for phonetic embeddings that leverages two linguistic phenomena, synonymy and homophony (§3). We also include phonetic and grapheme-based distractors based on edit-distance. (b) We use PSET to evaluate popular phonetic embeddings (Articulatory Embeddings, Phoneme2Vec, XPhoneBERT) and semantic embeddings (BERT, Word2Vec), see §4). The phonetic models perform reasonably well on this testbed; however, XPhoneBERT performs worse than the other two models. (c) We also use PSET as a prompt-based challenge to test five recent large language models, GPT-4o (OpenAI et al., 2024), Gemini 2.5 flash (Team et al., 2024), Llama 3.1-8B

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 7358–7368
November 4-9, 2025

(Dubey et al., 2024), OLMo-7B (Groeneveld et al., 2024) and OLMo 2-7B (OLMo et al., 2024) (§4.2). Gemini 2.5 Flash is the best performing model. The code and data to reproduce the experiments can be found on the following GitHub folder.

2 Related Work

The most relevant to our research is PWESuite (Zouhar et al., 2024), a benchmark based on phonetic tasks, such as rhyme and cognate detection. Although PWE is a valuable benchmark, it has limitations. In some cases, test creation is based on articulatory features and articulatory distance. Articulatory features can themselves be used to create phonetic embeddings of words; some of their tests thus favor, by design, certain types of embeddings. PSET, however, is not biased toward any type of embedding. Moreover, PWESuite tests whether embeddings encode phonetic information. In contrast, we test whether phonetic information takes precedence over semantic information.

Another approach was proposed by Kolachina and Magyar (2019) who created 12 artificial languages. Unlike our approach, their approach is mainly based on qualitative assessments. Efrat et al. (2023) developed a testbed for LLMs that tests elementary linguistic tasks, including recognizing homophones in word triplets. Some of their triplets contain automatically identified similar words as distractors. A similar approach was also proposed by Choi et al. (2024), who automatically selected word pairs (synonyms, near homophones, random words, same words and words spoken by the same speakers) and used these to evaluate self-supervised speech models (S3Ms). In both cases, the authors did not specifically focus on phonetic embeddings. Moreover, we create a more controlled and challenging testbed, by including manually curated synonyms and edit-distance based phonetic distractors.

3 The Testbed

Setup Our testbed is based on word quintets, with each quintet consisting of: (i) A reference word, referred to as the **anchor word (A)**; (ii) **a homophone (H)** of the anchor word; (iii) **a synonym (S)** of the anchor word, (iv) **A phonetic-based distractor word D_P** which is an IPA transcribed word with an **edit-distance** of 1 from the anchor and (v) **a grapheme-based distractor word D_G**, which has an edit-distance of 1 to the the anchor's spelling, thus the difference is purely based on graphemes without considering pronounciation. For example, "*ferry*" (anchor), "*fairy*" (homophone), "*boat*" (synonym), "*barie*" (phonetic-based distractor word) and "*berry*" (grapheme-based distractor word). Our setup using anchor words is inspired by a testbed to test style representations (Wegmann and Nguyen, 2021). Given an anchor word, a phonetic embedding model should assign the highest similarity to the homophone. Our final testbed contains 250 word quintets.

Homophone and synonym selection First, we compared several lists of homophones with each other. Our reasoning was that homophones that appear in several lists, are more likely to be common in the English language.[1] Second, we manually examined the words of each homophone pair to determine if they had synonyms using the Thesaurus dictionary, an online dictionary of synonyms[2]. We excluded pairs of homophones in which no word had synonyms (e.g., the pair *their, there*). Third, we used the Cambridge dictionary to ensure the synonym matched the anchor's meaning precisely, i.e., both used in similar contexts with near-identical meanings. For example, we excluded the term *read* because none of its synonyms, such as *see* and *study*, were direct synonyms to it. Details about the choice of dictionaries and synonyms are in Appendix A.1. We also checked the absence of semantic overlap between the homophone and the anchor word. For example, we excluded spelling variants, such as *archaeology* and *archeology*. Despite these steps, there are cases where the synonym does not match the meaning of the anchor word exactly, for example, *know* (anchor), *no* (homophone), *learn* (synonym), *beau* (phonetic-based distractor word), *enow* (grapheme-based distractor word). However, this is not a problem for our testing approach, as long as the anchor is clearly more semantically similar to the synonym than to the homophone.

Edit-distance distractors Our testbed includes two types of distractors: **phonetic-** and **grapheme-based**. We include phonetic distractors because the tested phonetic embedding models (i.e., not the LLMs) use phonetic transcriptions of text, such as

[1] The online list with homophone pairs, used as a starting point, was English Homophones (Gist), accessed 26/08/2025. Other homophone lists we accessed for comparison are English Homophones (Singularis), English Homophones (Oxford) and English Homophones (AllaboutLearning), accessed 26/08/2025.

[2] Thesaurus, accessed 26/08/2025.

IPA[3] or ARPAbet[4]. To create **phonetic distractors**, we first transcribed each word of a full English dictionary[5] and all our anchor words to IPA (more on the choice of IPA in Appendix B). Then, for each anchor word, we identified words from the dictionary with an edit-distance of 1. We filtered them using the NLTK word list (Bird et al., 2009) to remove uncommon entries. We then manually selected the word from the remaining set that sounded most similar to the anchor. Phonetic distractor words are phonetically close to the anchor word, while homophones are phonetically (almost) identical to it. A model that effectively captures phonetic relationships should associate the homophone more closely with the anchor word than the distractor. We furthermore include **grapheme-based distractors** to assess the extent to which a model relies on individual characters as opposed to phonetic features. The grapheme-based edit-distance distractors were created using the same procedure, but without the IPA transcription. All edit-distance words were also manually checked to ensure they are not synonymous with the anchor word. More details about the distractors are in Appendix B.

4 Experiments

We discuss two types of tests: one based on phonetic embeddings using cosine similarity (§4.1), the other based on prompting for LLMs (§4.2).

4.1 Testing Embeddings

Models For our experiments, we selected two semantic models and three phonetic models for embedding extraction. They represent different basic approaches to creating embeddings, allowing us to evaluate a variety of models for capturing semantic and phonetic information.

As phonetic models, we tested (a) **Articulatory embeddings (C.1)**, as first proposed by Mortensen et al. (2016). Articulatory embeddings for a word are composed of only 21 values, which are the average of the vectors of each IPA phoneme in the word. (b) **Phoneme2Vec (20 dimensions) (C.2)** (Fang et al., 2020), A version of Word2vec trained with ARPA symbols which represent the sounds of speech, and (c) **XPhoneBERT (88M param.)** (C.3) (The Nguyen et al., 2023), a language model based on IPA transcriptions. Its training style, which is the same as RoBERTa, could lead the model to also capture semantic and syntactic elements of the text. Note that XPhoneBERT was not initially trained for our purpose, but to enhance Text-To-Speech models. More details about the models are in Appendix C.

Even though we focus on evaluating phonetic embeddings, we also included two popular semantic embedding models for comparison. We expect their performance to be poorer. We included (a) **Word2vec** (Mikolov et al., 2013)[6], and (b) **BERT (110M param.)** in its Base version (Devlin et al., 2019). BERT and XPhoneBERT are contextual models. Static embeddings were extracted by averaging the last hidden layer of the target token across 10 sentences. More details about this process in Appendix C.4.

Using the testbed PSET can be used with any method that can assess the similarity between two words. For each quintet, we calculate: (i) sim(A,H), the similarity between the Anchor (e.g., '*feat*') and the Homophone (e.g., '*feet*'); (ii) sim(A,S), the similarity between the Anchor and the Synonym (e.g., '*triumph*'); (iii) sim(A, D_P), the similarity between the Anchor and the Phonetic-based Distractor (e.g., *beat*); and (iv) sim(A, D_G) the similarity between the Anchor and the Grapheme-based Distractor (e.g., *meat*). We use cosine similarity to measure the similarity. As phonetic embeddings should prioritize phonetic information, sim(A,H) should receive the highest similarity score for each quintet.

Models	H	S	D_P	D_G
Art. Phonemes (*phon*)	0.748	0.020	0.108	0.120
Phoneme2Vec (*phon*)	**0.903**	0	0.056	0.036
XPhoneBERT (*phon*)	0.730	0	0.170	0.090
Word2Vec (*sem*)	0.072	0.744	0.052	0.132
BERT (*sem*)	0.050	0.750	0.040	0.140

Table 1: Performance of phonetic (phon) and semantic (sem) models on each word category, showing how often each was rated most similar to the anchor. Bold highlights the best models for homophones and synonyms.

Results Our results (Table 1) show that all three phonetic models perform reasonably well on our test, since most of the time the homophone receives the highest similarity score. However, the

[3] the International Phonetic Alphabet.

[4] a phonetic alphabet developed by the Advanced Research Projects Agency (Carnegie-Mellon-University, 1993).

[5] List of English Words, accessed 26/08/2025

[6] We used the pre-trained *word2vec-google-news-300* model using the Gensim library.

Model	Technical Language (TL) Prompt					Layman's Language (LL) Prompt				
	H	S	D_P	D_G	err.	H	S	D_P	D_G	err.
GPT-4o	0.81 ± 0.02	0	0.05	0.07	0.06	0.78 ± 0.06	0.024	0.044	0.058	0.094
Gemini 2.5 Flash	$\mathbf{0.87 \pm 0}$	0	0.02	0.02	0.07	$\mathbf{0.87 \pm 0}$	0	0.02	0.03	0.06
Llama 3.1–8B	0.49 ± 0.12	0.04	0.07	0.22	0.17	0.34 ± 0.05	0.06	0.09	0.24	0.25
Olmo-7B	0 ± 0	0	0	0	1	0 ± 0	0	0	0	1
Olmo-2-7B	0.22 ± 0.04	0.30	0.08	0.16	0.21	0.17 ± 0.06	0.36	0.07	0.15	0.23

Table 2: LLM performance on homophones (H), synonyms (S), and distractors (D_P and D_G), showing how often each was rated most similar to the anchor. We also report extraction errors (err), with mean and standard deviation on homophone scores. Gemini 2.5 Flash performed the best.

results vary, and unexpectedly, **XPhoneBERT** is the model with the worst performance, despite being the most recent among those selected. Note that originally XPhoneBERT was developed for another goal (Text-To-Speech tasks), which could be one of the causes of its suboptimal performance. Both semantic models almost never choose homophones and tend to prefer synonyms, as expected. We also find that the frequency of words impacts performance, e.g., frequent phoneme-based edit-distance distractors are more likely to cause models to fail (see Appendix F).

4.2 Prompt-based Testing with LLMs

We also used PSET to test LLMs. Instead of directly applying PSET to the LLMs' embeddings, we use a prompt-based testing approach. This approach enables us to evaluate whether the LLMs have acquired phonetic information about words. Since we do not use their embeddings, the results are not directly comparable to embedding models.

Models We tested five LLMs of which two are closed and three are open-weight (Instruct versions): Gemini 2.5 Flash (Team et al., 2024), GPT4-o (OpenAI et al., 2024), Llama 3.1-8B (Dubey et al., 2024), OLMo-7B (Groeneveld et al., 2024) and OLMo 2-7B (OLMo et al., 2024).

Using the testbed We experimented with the LLMs using zero-shot prompting and two prompts. One uses a more **Technical language (TL)**: '*Which word is more **phonetically similar** to [ANCHOR]: [WORD1], [WORD2], [WORD3] or [WORD4]? Only respond with the correct word.*"; the other uses more **Layman's language (LL)**: "*Which word **sounds more** like [ANCHOR]: [WORD1], [WORD2], [WORD3] or [WORD4]? Only respond with the correct word.*". We ran each of the prompts twice, changing the order of the homophones, synonyms, and distractors (i.e., words 1–4). In total, each LLM is thus prompted 1000 times, 500 times for each of the prompt styles.

Results We report results for each prompt style separately (Table 2). For each style, the results are obtained by averaging the outcomes of two prompts with varied word orders. The LLMs sometimes made mistakes: they reproduced the anchor word, generated misspellings or did not provide any answer at all. These are reported under the "Extraction error (err.)" column. Gemini 2.5 Flash performed best, selecting the homophone in almost all cases (TL: 0.87, LL: 0.87), while GPT-4o stands in the second place (TL: 0.81, LL: 0.78). OLMO-7B fails to provide results in the correct format. OLMO 2-7B, on the other hand, tends in many cases to choose the synonym over the homophone. The results of the technical language prompt are higher for all LLMs, except OLMo-7B, which had difficulty with both prompts. Detailed results for the individual prompts are provided in Appendix E.

5 Conclusion

In our study, we introduce the PSET as a novel testbed designed to assess phonetic embeddings. PSET uses homophony, synonymy and edit-distance based distractors to evaluate whether embeddings prioritize phonetics over semantics. Using PSET, we tested three phonetic models and two semantic models representing different conceptual approaches. The most recent model, XPhoneBERT, performed the worst on PSET, followed by Articulatory Embeddings, while Phoneme2Vec appears to be the most effective. We also show that PSET can be used as a prompt-based test to assess whether LLMs can identify the word that sounds most similar to an anchor word. Out of the five LLMs we tested, Gemini 2.5 Flash performed the best.

Limitations

Our testbed has the following limitations. First, because we curated each instance in our dataset to ensure its quality, the dataset's final size is relatively small. Future work could explore expanding our dataset using automatic methods.

Second, our testbed is limited to English. However, many of our tested models, such as the articulatory embeddings, XPhoneBERT and the LLMs, already support many languages. Future work could focus on extending our testbed to other languages. However, not all languages have the same rate of homophony. Consequently, creating a similar dataset for some other languages can be more challenging.

Third, due to the criteria we imposed on the selection of words, we also had to include some low-frequency words. However, word frequency has an important impact on semantic models (Sahlgren and Lenci, 2016). For this reason, we have carried out an analysis of the frequency of words present in our quintets. Higher frequency distractors often lead to more errors, while higher frequency homophones increase correct identification in some models, with effects varying by embedding type. See Appendix, section F.

Fourth, our dataset also includes synonyms that do not always correspond to a word's most frequent meaning. This could reduce the likelihood of (semantics-focused) models selecting these synonyms, potentially simplifying the task. Nevertheless, we do not anticipate that this will substantially affect our overall conclusions.

Fifth, there are several potential confounding factors that may influence the structure of the embeddings. These confounding factors include Part-of-Speech, syntactic dependencies, and polysemy. For example, whether certain words share the same Part-of-Speech may affect the embedding results.

Finally, we cannot rule out the possibility that some of the language models may have been exposed to the word lists used in our paper. However, we note that their performance was far from perfect, with some LLMs performing quite poorly.

Acknowledgments

We are grateful to the reviewers for their insightful feedback, which has significantly improved the quality of this paper. Dong Nguyen is funded by the Veni research programme with project number VI.Veni.192.130, which is (partly) financed by the Dutch Research Council (NWO). Gianluca Sperduti has been funded by the WEMB project (B53D23013050006). Funded by the European Union — Next Generation EU.

References

Khuyagbaatar Batsuren, Gabor Bella, and Fausto Giunchiglia. 2019. CogNet: A large-scale cognate database. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3136–3145, Florence, Italy. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4758–4781.

Carnegie-Mellon-University. 1993. The CMU pronouncing dictionary. `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`. Accessed: 2025-05-14.

Kwanghee Choi, Ankita Pasad, Tomohiko Nakamura, Satoru Fukayama, Karen Livescu, and Shinji Watanabe. 2024. Self-supervised speech representations are more phonetic than semantic. In *Interspeech 2024*, pages 4578–4582.

Mansi Chugh, Peter A. Whigham, and Grant Dick. 2018. Stability of word embeddings using word2vec. In *AI 2018: Advances in Artificial Intelligence - 31st Australasian Joint Conference, Wellington, New Zealand, December 11-14, 2018, Proceedings*, volume 11320 of *Lecture Notes in Computer Science*, pages 812–818. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, and other 98 authors. 2024. The Llama 3 herd of models. *CoRR*, abs/2407.21783.

Avia Efrat, Or Honovich, and Omer Levy. 2023. LMentry: A language model benchmark of elementary language tasks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10476–10501, Toronto, Canada. Association for Computational Linguistics.

Anjie Fang, Simone Filice, Nut Limsopatham, and Oleg Rokhlenko. 2020. 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*, pages 699–708.

Dirk Groeneveld, Iz Beltagy, and other 41 authors. 2024. Olmo: Accelerating the science of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15789–15809. Association for Computational Linguistics.

Francesca Incitti, Federico Urli, and Lauro Snidaro. 2023. Beyond word embeddings: A survey. *Inf. Fusion*, 89:418–436.

Sudheer Kolachina and Lilla Magyar. 2019. What do phone embeddings learn about phonology? In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 160–169, Florence, Italy.

Hairong Liu, Mingbo Ma, Liang Huang, Hao Xiong, and Zhongjun He. 2019. Robust neural machine translation with joint textual and phonetic embedding. pages 3044–3049. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

David R. Mortensen, Patrick Littell, Akash Bharadwaj, Kartik Goyal, Chris Dyer, and Lori Levin. 2016. PanPhon: A resource for mapping IPA segments to articulatory feature vectors. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3475–3484, Osaka, Japan. The COLING 2016 Organizing Committee.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. 2 olmo 2 furious. *Preprint*, arXiv:2501.00656.

OpenAI, Josh Achiam, and 283 other authors. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

Allison Parrish. 2017. Poetic sound similarity vectors using phonetic features. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 13, 2, pages 99–106.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).

Magnus Sahlgren and Alessandro Lenci. 2016. The effects of data size and frequency range on distributional semantic models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 975–980. The Association for Computational Linguistics.

Miikka P. Silfverberg, Lingshuang Mao, and Mans Hulden. 2018. Sound analogies with phoneme embeddings. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2018*, pages 136–144, Salt Lake City, US.

Gemini Team, Rohan Anil, and 1376 other authors. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Linh The Nguyen, Thinh Pham, and Dat Quoc Nguyen. 2023. XPhoneBERT: A Pre-trained Multilingual Model for Phoneme Representations for Text-to-Speech. In *Proc. INTERSPEECH 2023*, pages 5506–5510.

Walter J. B. van Heuven, Pawel Mandera, Emmanuel Keuleers, and Marc Brysbaert. 2014. Subtlex-uk: A new and improved word frequency database for british english. *Quarterly Journal of Experimental Psychology*, 67(6):1176–1190. PMID: 24417251.

Anna Wegmann and Dong Nguyen. 2021. Does it capture STEL? A modular, similarity-based linguistic style evaluation framework. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7109–7130, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Jian Zhu, Cong Zhang, and David Jurgens. 2022. Byt5 model for massively multilingual grapheme-to-phoneme conversion.

Vilém Zouhar, Kalvin Chang, Chenxuan Cui, Nate B. Carlson, Nathaniel Romney Robinson, Mrinmaya Sachan, and David R. Mortensen. 2024. PWESuite: Phonetic word embeddings and tasks they facilitate. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13344–13355, Torino, Italia. ELRA and ICCL.

A Synonym selection

A.1 Dictionaries

Dictionaries The Thesaurus dictionary is available at Thesaurus, and the Cambridge Dictionary can be accessed at Cambridge dictionary. The Thesaurus provides synonyms for a word without defining its meaning. Cambridge, on the other hand, lists all the meanings of a word. We checked the Cambridge dictionary to ensure that at least one of the meanings of the selected synonyms matched strongly with that of the anchor word.

Synonyms We excluded those words where the synonymy relationship could be ambiguous, such as "read" and "see" [7]. "I read a paper" and "I saw a paper" can be used in similar semantic contexts. However, "see" and "read," if looked up in English dictionaries, mean profoundly different things. In short, we aimed to reduce the impact of synonyms whose similarity depends solely on the context, rather than having a true general synonymy. In those cases, we preferred to proceed with a word with a similar semantic sphere, but with a less ambiguous semantic relationship.

B Edit-distance details

Choice of IPA We used IPA (rather than ARPAbet) for distractor construction because it's widely supported by automatic transcription tools and aligns with two of the three phonetic models we evaluate. The transcription tool we used is CharsiuG2P (Zhu et al., 2022), the same tool used by the authors of XPhoneBERT. The model can be accessed from HuggingFace at the following URL: Charsiu G2P.

Distractor Exceptions Sometimes, it was not possible to find distractors with an edit distance of 1 from the anchor word. To address this, we allowed minor morphological variations, such as changes in singular/plural forms, for a small subset of distractors—specifically, 8 phonetic-based and 18 grapheme-based distractors. For example, the anchor word principal might have principals as its phonetic-based distractor. This workaround affects only a limited number of quintets and does not compromise the integrity of the task, as the correct answer remains unchanged. Additionally, some distractors may belong to specific dialects or take the form of exclamations. This does not impact the dataset's quality, since distractors are intended for use with phonetic models. For instance, in the quintet new (simplified pronunciation: nu), knew (nu), recent (ricent), boo (bu), and anew (aenu), the word boo serves as an exclamation commonly used to mimic a shout or to startle someone.

[7] The words of this example are taken directly from the synonymy dictionary Thesaurus: Example for the word *read* from the Thesaurus Dictionary, accessed 26/08/2025.

C Models

C.1 Articulatory embeddings

Articulatory Embeddings were first proposed by Mortensen et al. (2016). First, each symbol of the International Phonetic Alphabet (IPA) is represented as a 21-size vector, which indicates whether a particular phonetic feature is present, absent, or irrelevant in the production of a sound. For example, if a sound involves the use of the lips, it is assigned the value +1; if the lips are not involved, the value -1 is assigned; if the characteristic is irrelevant, the value 0 is assigned. To obtain the embeddings of a word, we take the average of all articulatory vectors extracted from its IPA symbols. Other articulatory embeddings approaches were proposed by Silfverberg et al. (2018) and Kolachina and Magyar (2019).

C.2 Phoneme2Vec

Phoneme2Vec is a direct transposition of Word2Vec for phonetics. The goal is to predict the phonemes that are in the vicinity of a given phoneme. Although Fang et al. (2020) propose two more complex versions, we chose the simplest method to represent this idea.

We trained this model using ARPAbet and the CMU Pronunciation Dictionary. ARPAbet is a phonetic transcription system primarily used for English. It uses ASCII sequences to represent phonemes and allophones. The CMU (Carnegie Mellon University) pronunciation dictionary includes approximately 134k instances manually mapped from grapheme to ARPAbet. Thus, each word is represented as a sequence of phonemes. These sequences serve as the "corpus" for training the Phoneme2Vec model. In this case, therefore, each context is represented by the ARPAbet phonemes of the word itself in the dictionary, not by a sentence. The purpose of the training is to create vector representations for individual ARPAbet phonemes by using their surrounding phonemes as context. By doing this across multiple words, the Word2Vec model learns meaningful relationships

between phonemes based on how they are used together in different words. In our experiment, we used standard hyperparameters: a vector size of 20, a context window of 2, and the Skip-Gram model (*sg=1*). For our test, we excluded quintets that had words not present in the CMU.

C.3 XPhoneBERT

XPhoneBERT (The Nguyen et al., 2023) is a language model that captures phonetic features. Even though this language model was trained to improve other models on Text-To-Speech tasks, we use it to extract phonetic embeddings. The model was trained with the RoBERTa pre-training approach and used BERT$_{Base}$ as an architecture. Specifically, The Nguyen et al. (2023) transcribed all the BERT datasets into IPA in the various languages chosen. They based their training on tokens of words transcribed into IPA, with a maximum of 512 tokens for each sentence considered. Note that originally XPhoneBERT was developed for another goal (Text-To-Speech tasks), which could be one of the causes of its suboptimal performance.

C.4 Extraction of static embeddings from contextual models

To extract static embeddings using contextual models (BERT, XPhoneBERT) we applied an approach similar to the one described in Bommasani et al. (2020). Specifically, for each target word in the quintets, we extracted 10 sentences from Wikitext.[8] If fewer than 10 sentences were available for a word in Wikitext, we used ChatGPT to generate additional sentences. We extracted the last hidden layer of the target token for each of the 10 different sentences. We then used the average across all 10 sentences as our contextual embedding.

D PWE Suite

We also tested the three embeddings models (Articulatory Phonemes, Phoneme2Vec, XPhoneBERT) with the recently proposed testbed for phonetic embeddings, PWESuite (Zouhar et al., 2024). The goal is to understand if there are any differences, and which ones, between our testbed and PWESuite. Although some of the models we reported are already tested in the original PWESuite paper, we re-ran the experiments from scratch to have a directly comparable result in the same setting. Compared to PSET, in PWESuite the articulatory embeddings perform better. XPhoneBERT also performs the worst among the three on this testbed.

[8] WikiText

E Detailed LLMs results

We analyzed the performance of five language models using different prompt variants and word orders (see also Section 4.2). The models are: GPT-4o, Gemini 2.5 Flash, Llama 3.1-8B, OLMo-7B and OLMo 2-7B. We experimented with a prompt in more technical language and varied the order of the words (i.e., homophones, synonyms, distractors), resulting in technical prompts 1 and 2. Similarly, we also varied the word order of the layman's prompt, resulting in layman's prompt 1 and 2. The results of the technical language prompt are higher for all LLMs, except OLMo-7B, which had difficulty with both prompts. Tables 5, 6, 7 and 8 show the performance metrics for each single prompt for all the models.

P.	H	S	D_P	D_G	err.
Tech. (1)	0.796	0.000	0.056	0.076	0.072
Tech. (2)	0.832	0.000	0.044	0.064	0.060
Lay. (1)	0.776	0.028	0.040	0.064	0.092
Lay. (2)	0.784	0.020	0.048	0.052	0.096
Average	0.797	0.012	0.047	0.064	0.080

Table 5: Performance of GPT4-o per prompt. For each category (**H**omophones, **S**ynonyms, **D**istractors), we report the proportion of cases where it was assigned the highest similarity to the anchor word. We also report how often an extraction error (err.) occurred.
aliases: P. = Prompt, Tech. = Technical, Lay. = Layman, (1) = prompt number 1, (2) = prompt number 2.

P.	H	S	D_P	D_G	err.
Tech. (1)	0.864	0.0	0.036	0.028	0.072
Tech. (2)	0.884	0.0	0.016	0.024	0.076
Lay. (1)	0.868	0.0	0.032	0.036	0.064
Lay. (2)	0.872	0.0	0.024	0.036	0.068
Average	0.872	0.0	0.027	0.031	0.070

Table 6: Performance of Gemini per prompt.

P.	H	S	D_P	D_G	err.
Tech. (1)	0.576	0.064	0.060	0.14	0.16
Tech. (2)	0.404	0.020	0.088	0.304	0.184
Lay. (1)	0.268	0.104	0.104	0.180	0.344
Lay. (2)	0.416	0.020	0.092	0.308	0.164
Average	0.416	0.052	0.086	0.233	0.213

Table 7: Performance of Llama3.1-8B per prompt.

Model	**Human Sim.** (Pearson)	**Art. Dist.** (Pearson)	**Retrieval** (rank perc.)
Articulatory Phonemes	**0.60**	0.11	**0.84**
Phoneme2Vec	**0.60**	0.06	0.74
XPhoneBERT	0.14	**0.14**	0.66

Table 3: Results with PWESuite: Intrinsic evaluation of embedding methods.

Model	**Analogies** (Acc@1)	**Rhyme** (accuracy)	**Cognate** (accuracy)	**Overall**
Articulatory Phonemes	**0.99**	0.85	**0.62**	**0.67**
Phoneme2Vec	0.31	**0.86**	0.60	0.53
XPhoneBERT	0.09	0.59	0.60	0.37

Table 4: Results with PWESuite: Extrinsic evaluation and overall performance of embedding methods.

P.	H	S	D_P	D_G	err.
Tech. (1)	0.04	0.04	0.0	0.0	0.992
Tech. (2)	0.0	0.0	0.0	0.0	1.000
Lay. (1)	0.08	0.012	0.0	0.0	0.98
Lay. (2)	0.0	0.0	0.0	0.04	0.996
Average	0.003	0.001	0	0.001	0.992

Table 8: Performance of OLMo-7B per prompt.

P.	H	S	Dp	D_G	err.
Tech. (1)	0.26	0.392	0.116	0.048	0.184
Tech. (2)	0.192	0.216	0.060	0.280	0.252
Lay. (1)	0.124	0.536	0.060	0.028	0.252
Lay. (2)	0.216	0.196	0.092	0.272	0.224
Average	0.198	0.335	0.082	0.157	0.228

Table 9: Performance of OLMo 2-7B per prompt.

F Frequency Analysis

The frequency of words influences the representational stability of embeddings (Chugh et al., 2018). To investigate the potential influence of word frequency on correctly identifying the homophone, we conducted a frequency analysis on the quintets from our dataset. We used the Subtlex-UK frequency dataset for reference (van Heuven et al., 2014), specifically focusing on the Zipf scale metric. The Zipf scale is a logarithmic measure that ranks words on a scale from 1 (infrequent) to 7 (very frequent), corresponding to log10 (frequency per billion words). We fitted a logistic regression model. Our dependent variable is whether the model correctly predicted the homophone (Class 1) or not (Class 0). The independent variables are the Zipf scale frequencies of the classes. Table 10 shows the coefficients of each independent variable and the intercept, for each model. Across all models, a higher frequency in Phonetic-based edit-distance distractors (Edit-Distance P.) increases the likelihood of incorrectly predicting the homophone (Edit-Distance P., Edit-distance G., Synonym). The opposite holds for the Edit-distance G. class, where higher frequency increases the likelihood of predicting class 1 (Homophone), except for Word2Vec.

Furthermore, in our testbed, anchors and homophones (Section 3) have very different meanings. For this reason, when the words are semantically well-represented, it is expected for the semantic models to select Class 0 (not a homophone). The coefficients for Word2Vec show that both a higher frequency of a homophone and a higher frequency of a synonym reduce the probability of selecting the homophone. The result for BERT might instead be related to its contextual training with rather large datasets, which might have homophones close to each other in some cases (e.g. Wikipedia is part of the BERT's training set, Wikipedia page on homophones).

G Implementation details

For both XPhoneBERT and BERT, we used the Transformers (version 4.45) implementation from HuggingFace (Wolf et al., 2019). The same library was also used to query open-weight LLMs. All the LLMs were tested in their Instruct version. To query proprietary LLMs we referred to the APIs (Gemini: Gemini API, OpenAI: OpenAI API). Phoneme2Vec was trained using the Word2Vec's gensim framework (Rehurek and Sojka, 2011). For Word2Vec, we used the Google News pretrained version, which contains 300-dimensional vectors for 3 million words and phrases. The pretrained Google News Word2Vec model was accessed within the Gensim library[9]. The Articulatory Embeddings were extracted using the Pan-

[9]Gensim

Model	Anchor	Homophone	Synonym	Edit-distance P.	Edit-distance G.	Intercept
XPhoneBERT (*phon*)	0.089	0.054	0.151	**-0.183**	0.078	0.138
Phoneme2Vec (*phon*)	-0.256	-0.221	**-0.567**	-0.177	0.099	6.699
Art. Phonemes (*phon*)	0.308	0.183	-0.009	-0.289	0.527	-0.098
Word2Vec (*sem*)	-0.174	-0.397	-0.325	-0.072	-0.254	1.597
BERT (*sem*)	0.087	0.029	0.220	-0.085	0.034	-4.305

Table 10: The effect of word frequency. The table displays the coefficients and the intercept term for five different models: XPhoneBERT, Phoneme2Vec, Articulatory Phonemes (phonological models), Word2Vec, and BERT (semantic models). Significant coefficients are marked in bold.

Phon's library at the following URL: PanPhon. All the links were accessed on 26/08/2025. All the experiments were run using an NVIDIA A40. The extraction of the static version for BERT and XPhoneBERT embeddings required ~6 hours. The prompt extraction required ~2 hours. All other experiments were quickly run on the CPU.

H Copyright

This dataset is available under the Creative Commons Attribution 4.0 International License (CC BY 4.0). You are free to use, share, and adapt the data with appropriate credit to the author.

I Qualitative Error Analysis

LLMs To explore the nature of incorrect responses, we conducted a qualitative analysis of errors for each LLM. For each, three prompts resulting in incorrect answers were randomly selected. The goal was not exhaustive evaluation but a light diagnostic to identify any notable tendencies. Tables 11–15 list these prompts, the model outputs, and correct answers. Note that all the distractors shown follow the order: (H), (S), (D_P), (D_G). For a subset of examples, we visualized the relationship between anchors and distractors by extracting static embeddings for the open-source models (Olmo-7B, Olmo-2-7B, LLama3.1-8B) with the same method used for BERT and XPhoneBERT (see section C.4). Selected embeddings plots are shown in Figures 1–3. These figures highlight the models' semantic tendencies: without prompting, the extracted embeddings cluster synonyms rather than other distractors closer to the anchor, such as cash and money (Fig. 2), and pain and agony (Fig. 3).

Embedding models We also highlight two quintets where Phoneme2Vec and Word2Vec fail (Figures 4–5): *file (A), phial (H), record (S), bile (D_P), mile (D_G)* for Phoneme2Vec and *border (A), boarder (H), edge (S), balder (D_P), birder (D_G)* for Word2Vec. In the Phoneme2Vec case, errors emerge when the homophone and the anchor ends to be represented differently —e.g., *file–phial*, where the word *phial* was probably represented with its American and not British pronunciation, leading to an error. For Word2Vec, the failure mode is less transparent; a plausible cause is corpus-driven bias (e.g., frequency or co-occurrence artifacts) rather than phonetic proximity.

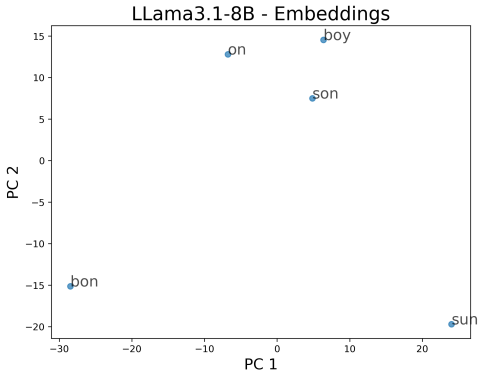

Figure 1: Embeddings for an incorrectly answered quintet - Olmo. Quintet: son (A), sun (H), boy (S), bon (D_P), on (D_G).

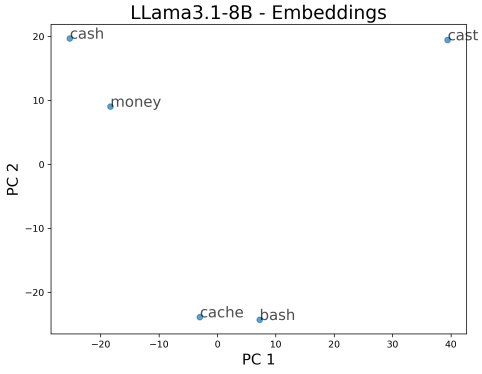

Figure 2: Embeddings for an incorrectly answered quintet - Olmo2. Quintet: cash (A), cache (H), money (S), bash (D_P), cast (D_G).

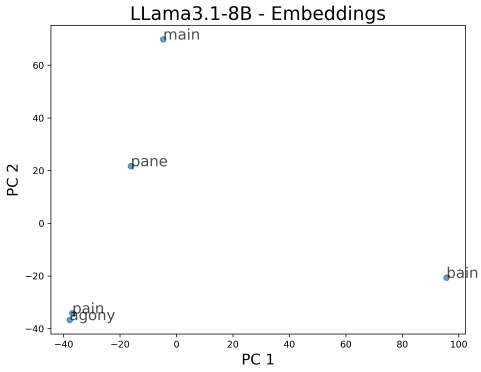

Figure 3: Embeddings for an incorrectly answered quintet - Llama3-8B. Quintet: pain (A), pane (H), agony (S), bain (D_P), main (D_G).

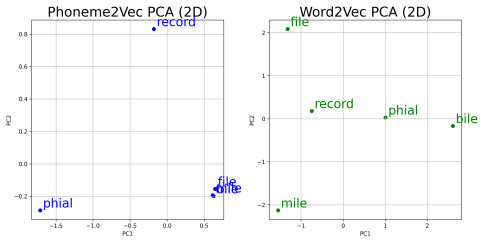

Figure 4: Embeddings of a quintet for which Phoneme2Vec produced an incorrect result. We also show the Word2vec embeddings.

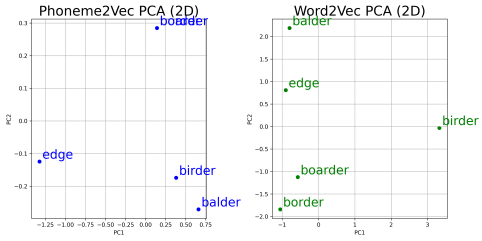

Figure 5: Embeddings of a quintet for which Word2Vec produced an incorrect result. We also show the Phoneme2Vec embeddings. Only four points are displayed, since two of the words have identical embeddings under Phoneme2Vec.

Gemini 2.5 Flash				
Prompt style	**Anchor**	**Distractors**	**Given Answer**	**Correct Answer**
TL	close	clothes, near, cloak, lose	cloak	clothes
LL	ferry	fairy, boat, barie, berry	berry	fairy
TL	vile	vial, bad, bile, aile	bile	vial

Table 11: Example prompt results from Gemini 2.5 Flash. All the distractors shown in the examples follow the order: (H), (S), (D_P), (D_G).

GPT-4o				
Prompt style	**Anchor**	**Distractors**	**Given Answer**	**Correct Answer**
TL	cash	cache, money, bash, cast	bash	cache
TL	rest	wrest, vacation, best, cest	best	wrest
TL	lead	led, guide, bead, read	bead	led

Table 12: Example prompt results from GPT-4o

Llama 3.1-8B				
Prompt style	**Anchor**	**Distractors**	**Given Answer**	**Correct Answer**
TL	earn	urn, acquire, an, arn	arn	urn
LL	pain	pane, agony, bain, main	bain	pane
LL	sail	sale, cruise, bail, ail	ail	sale

Table 13: Example prompt results from Llama 3.1-8B.

Olmo-7B				
Prompt style	**Anchor**	**Distractors**	**Given Answer**	**Correct Answer**
LL	son	sun, boy, bon, on	boy	sun
TL	faint	feint, vague, fent, paint	faint feint,	feint
TL	cruise	crews, sailing, bruise, bruiser	cruise	crews

Table 14: Example prompt results from Olmo-7B.

Olmo-2-7B				
Prompt style	**Anchor**	**Distractors**	**Given Answer**	**Correct Answer**
TL	cash	cache, money, bash , cast	cast	cache (H)
TL	bait	bate, torment, babe, ait	ait	bate (H)
TL	rude	rued, blunt, bood, crude,	blunt	rued

Table 15: Example prompt results from Olmo-2-7B.