Rethinking Word Similarity: Semantic Similarity through Classification Confusion

Anonymous ACL submission

Abstract

Word similarity is important for NLP and its applications to humanistic and social science tasks, like measuring meaning changes over time, detecting biases, understanding contested terms, and more. Yet the traditional similarity method based on cosine between word embeddings falls short in capturing the context-dependent, asymmetrical, polysemous nature of semantic similarity.

We propose a cognitively-inspired model drawing on the proposal of Tversky (1977) that for conceptual tasks, people focus on extracting and compiling only the relevant features. Our `Word Confusion` model reframes semantic similarity in terms of feature-based *classification confusion*. We train a classifier to map from contextual embeddings to words and use the classifier confusion (the probability of choosing confound word c instead of correct target t) as a measure of the similarity of c and t.

We show that `Word Confusion` outperforms cosine similarity in matching human similarity judgments across several datasets (MEN, WirdSim353, and SimLex), can measure similarity using predetermined features of interest, and enables qualitative analysis on real-world data. Reframing similarity based on classification confusion offers a cognitively-inspired, directional, and interpretable way of modeling the relationship between concepts.

1 Introduction

Semantic similarity measures allow computational social scientists, digital humanists, and NLP practitioners to perform fine-grained synchronic and diachronic analysis on word meaning, with important applications to areas like cultural analytics and legal and historical document analysis (Bhattacharya et al., 2020; Ríos et al., 2012).

The cosine between two embedding vectors is the most commonly used similarity metric for textual analysis across a variety of fields, including the digital humanities (Johri et al., 2011; Caliskan et al., 2017; Manzini et al., 2019; Martinc et al., 2020). However, it does not fully account for the multi-faceted nature of similarity (Tversky, 1977; Ettinger and Linzen, 2016; Zhou et al., 2022a, inter alia). Cosine similarity is dominated by a small number of rogue dimensions due to the anisotropy of contextual embedding spaces (Timkey and Van Schijndel, 2021), underestimates the semantic similarity of high-frequency words (Zhou et al., 2022a), is a symmetric metric that cannot capture the asymmetry of semantic relationships[1] (Vilnis and McCallum, 2014), and often fails in capturing human interpretation (Sitikhu et al., 2019). These make cosine similarity less than optimal as a tool for humanistic and social scientific analytics.

Here we propose to think about concept similarity metrics in a different way, inspired by Tversky's 1977 seminal work on similarity. Such cognitive models presume that humans have a very rich mental representation of concepts. When faced with a particular task, like similarity assessment, we extract and compile from this rich representation only the relevant features for the required task. This formulation highlights the context-dependency of similarity judgments (Evers and Lakens, 2014).

To demonstrate the potential of this new framing, we introduce a proof-of-concept: `Word Confusion`, a self-supervised method the **defines the semantic similarity between words according to a classifier's confusion between them**. In our new model, we first train a classifier to map from a word embedding to the word itself, distinguishing it from a set of distractors. At inference time, given a new embedding e for a target word t, the probability the classifier assigns to a confound word c, is used as

[1] Human similarity judgments are directional; "cat" is more similar to "animal" than "animal" is to "cat".

a measure of similarity of words c and t. The set of distractor words used in training act as *features*, allowing the similarity between words to be based on their feature-based interchangeability.

We first test our model by comparing it to cosine in standard word-similarity tasks, and testing it in feature classification tasks like sentiment and grammatical gender classification. Our findings suggest that the classification errors by Word Confusion might serve as a meaningful metric for assessing the similarity between two words.

We then apply Word Confusion to two different data exploration tasks. We first validate Word Confusion on a real-life dataset by tracing how the dollar token "$" has changed over the years.

We next use Word Confusion to study a question in the political history of revolutionary France: how "revolution" went from being seen as a means of popular liberation, to becoming identified with governmental actions that often flouted such personal freedoms. We do this by measuring the Word Confusion similarity of the French word "revolution" to different sets of words in the French *Archive Parlementaires* from 1789-1793.

Our contributions are:

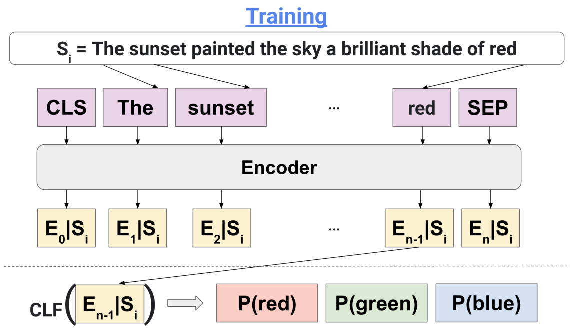
- We propose a novel framing of semantic similarity, inspired by cognitive models and sensitive to the pitfalls of cosine similarity. Our new formulation can learn more complex word identity boundaries than cosine similarity alone; accounts for the asymmetrical nature of semantic similarity; can be easily adapted to desired domains; and provides a more interpretable measure.

- We implement a proof-of-concept of our new framing of similarity, showing it outperforms cosine on standard semantic similarity benchmarks.

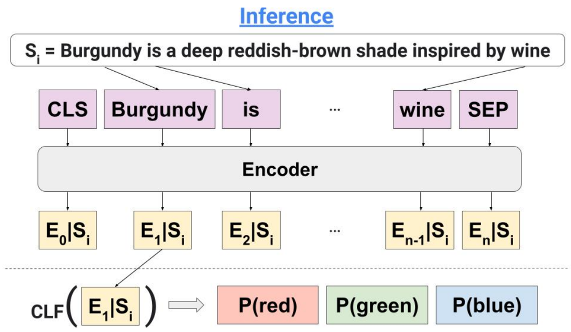
- We apply our method to real-world data, showcasing its potential for analyzing word meaning and temporal trends.

We hope this new formulation will spark the creation of computational social science tools that account for the multi-faceted and complex nature of semantic similarity[2].

[2]The Python package for this tool will be linked here upon paper acceptance.



(a) Training Word Confusion: The classifier is trained in a self-supervised manner. After constructing the desired features /classes of the classifier, we automatically extract sentences containing the feature words (red, green, and blue). The input to the classifier is the contextual embedding of the primary color token, e.g., the BERT embedding of the word "red" in conditioned on the sentence "The sunset painted the sky a brilliant shade of red". The classifier is trained to map between contextual embedding to the word.



(b) Word Confusion inference: The predetermined classes serve as inference-features. The input is a sentence with a word we wish to inspect, e.g., "burgundy". The trained classifier receives as input the contextual embedding "burgundy". We then use the classifier's confusion matrix to define the similarity of the burgundy with each and every primary color. We note that the input word at inference could be out-of-vocabulary with respect to the classifier. Moreover, a different set of classes will entail different features used to describe the input word.

Figure 1: Word Confusion: We predetermined a set of classes for our classifier. At training, we extract sentences containing the chosen classes {red, green, blue}. We then use BERT's contextual embeddings of these words to train the classifier to correctly map from the embeddings to the right class /feature (color, in this case). At inference, we extract BERT's contextual embeddings of a new word, that is not necessarily represented by a classifier class ("burgundy"). We then input the embedding to the classifier and use its confusion matrix to understand which primary colors burgundy is similar to.

2 Introducing Word Confusion

Figure 1 depicts Word Confusion's training and inference processes. At training, we predefined a set of words, or features, that will later be used to describe the analyzed word. We then extract from a corpus a set of sentences containing these words, such as “The sunset painted the sky a brilliant shade of red” for the word “red”.[3] We then use BERT to extract the contextual embeddings of these feature-words, and train a classifier to map from a word embedding to its corresponding word identity. Thus, the classifier’s training objective is to correctly classify the embedding to the word that corresponds to it.

More formally, given embeddings $\{e_1, e_2 .... e_i\} \in E$ that correspond to word identities $\{w_1, w_2, ..., w_i\} \in W$, where W is the chosen set of words, we train a logistic regression classifier on all pairs of $\{e_i, w_i\}$.

At inference, we wish to define the semantic similarity of a word in terms of the classifier’s classes (which can be thought of as features).[4] We extract the contextual embedding of the word we wish to inspect, e.g., the word “burgundy” given the sentence “Burgundy is a deep reddish-brown shade inspired by wine”. We use the trained classifier to map the “burgundy”-embedding to its classes, or features, which are in this case the primary colors. We then use the classifier’s confusion matrix to understand which primary colors burgundy is similar to. Similar to the chosen example, the input word at inference could be out-of-vocabulary with respect to the classifier. This method also works for the case in which the inspected word is one of the classifier’s classes, as we can ignore the probability it assigns to that word and use the other $N - 1$ features.

More formally, we use the probability distribution predicted by the model, $\vec{p_j} \in \mathbb{R}^{|W|}$, to quantify the semantic similarity between w_j (Burgundy) and $w_i, \forall w_i \in W = \{red,\ green,\ blue\}$. For example, the similarity of burgundy with the color red is the probability our classifier assigns to the class “red”. Thus the set of distractor words chosen to train the initial classifier act as features that can be selected by the analyst to focus on a particular dimension or question.

2.1 Benchmarking `Word Confusion`

The intuition behind `Word Confusion` is that if it struggles to distinguish between contextual embeddings of *burgundy* and *red*, this could indicate they are similar. To test this hypothesis, we use `Word Confusion` on three semantic similarity benchmarks. For each task, we trained a `Word Confusion` model using sentences from English Wikipedia[5]. Our classes contained all the words from the benchmark. We then built word embeddings by averaging the last four hidden layers of BERT-base-cased (additional details in appendix B).

To calculate the similarity between two words w_i, w_j, we first extracted all the sentences containing w_i from English Wikipedia. We averaged the contextual token embeddings of w_i using these sentences. This average token embedding was the input to the trained classifier (with classes containing all the words in the benchmark). We then used the probability `Word Confusion` assigned to w_i as the right class to set the similarity score between w_i and w_j. We used three benchmarks:

- **MEN** contains 3000-word pairs annotated by 50 humans based on their “relatedness” (Agirre et al., 2009). For example {berry, seed}, {game, hockey}, and {truck, vehicle} received high relatedness scores, where {hot, zombi}, {interior, mushroom}, and {bakery, zebra} received low scores. To approximate human agreement, two annotators labeled all 3000 pairs on a 1-7 Likert scale; their Spearman correlation is 0.68, and the correlation of their average ratings with the general MEN scores is 0.84.
- **WordSim353 (WS353)** contains 2000 word-pairs along with human-assigned association judgements (Bruni et al., 2014). For example {bank, money}, {Jerusalem, Israel}, and {Maradona, football} received high scores whereas {noon, string}, {sugar, approach}, and {professor, cucumber} were ranked low. The authors report an inter-annotator agreement of 84%.
- **SimLex** contains 1000 word-pairs and directly measures similarity, rather than relatedness or association (Hill et al., 2015). The authors defined similarity as synonymy and instructed their annotators to rank accordingly. For example {happy, glad}, {fee, payment}, and {wisdom, intelligence} received high relatedness scores, where {door, floor}, {trick, size}, and {old, new} received low scores. Inter-rater agreement (the average of pairwise Spearman correlations between the ratings of all respondents) was reported as 0.67.

[3] We use at least 30 training examples per class.

[4] We note that a different set of classes will entail different features used to describe the input word.

[5] We use at least 30 training examples per class.

Method \ Dataset	MEN	WS353	SimLex
Cosine	0.68	0.55	0.52
`Word Confusion`	**0.76**	**0.69**	**0.60**

Table 1: Spearman's ρ correlation between `Word Confusion` and cosine similarity results as compared to humans. These three benchmarks focus on slightly different aspects of word similarity. We measure the correlation between human scores and cosine similarity between the language model embeddings versus `Word Confusion`'s similarity scores. As can be seen, our method outperforms cosine similarity.

Across MEN, WS353, and SimLex, `Word Confusion` outperforms cosine similarity, with Spearman's ρ that are up to 0.14 higher (see Table 1). This illustrates the meaningfulness of classification confusions, compared to cosine similarity. We note that our probability distribution spanned only the classes we chose in advance (all of the words in the dataset), which yields a different vocabulary compared to the original language model.

3 Theoretical Intuition

In this section, we discuss the importance of word identifiability and how it enables the core mechanics of `Word Confusion`. We then discuss the theoretical differences between `Word Confusion` and cosine similarity.

3.1 The Identifiability of Contextualized Word Embeddings

`Word Confusion` depends on the ability of a classifier to identify a word based on its contextual embedding; here we confirm that this classification task is indeed solvable, and examine some error cases to better understand it.

While contextualized word embeddings vary in their representation based on context, prior work showed that tokens of the same word still cluster together in geometric space (Zhou et al., 2022b).

To test whether these boundaries are indeed learnable, we test how well a model can identify a contextualized word embedding after seeing one other example of the same word's contextualized embedding. We randomly sampled 26,000 words from English Wikipedia, trained 1000-class one-shot classifiers, and tested them on 10,000 examples (ten examples per class). Indeed, we found that the average test set accuracy on all our classifiers is 90%, suggesting that the contextualized word embeddings are highly *identifiable*. Thus, given an embedding, it is possible to identify its symbolic representation. See appendix A for additional experimental details.

3.2 Theoretical Differences Between `Word Confusion` and Cosine Similarity

We now discuss the theoretical differences between `Word Confusion` and cosine similarity, arguing that feature-based similarity can produce more flexible decision boundaries, capture asymmetrical relations, highlight specific aspects of the analyzed word, and output more meaningful scores.

Decision Boundaries. We now provide some theoretical intuition behind why using logistic regression to predict the identity of embeddings differs from the commonly used cosine metric.

Given two normalized vectors in 2-dimensions, x and y, we apply a linear transformation A to each. Assuming A is real, the singular value decomposition of A is $U\Sigma V^\intercal$; thus we can rewrite Ax, Ay using the singular values of A: $\sigma_1 u_1 {v_1}^\intercal x_1 + \sigma_2 u_2 {v_2}^\intercal x_2$ and $\sigma_1 u_1 {v_1}^\intercal y_1 + \sigma_2 u_2 {v_2}^\intercal y_2$.

Depending on A, the distance between the two vectors after the linear transformation can be either bigger or smaller than the distance between the original vectors. E.g., the cosine distance between the projected vectors is ${\sigma_1}^2({v_1}^\intercal x_1)({v_1}^\intercal y_1) + {\sigma_2}^2({v_2}^\intercal x_2)({v_2}^\intercal y_2)$ compared to $1 - (x_1 y_1 + x_2 y_2)$. Similarly, the Euclidean distance between the project vectors is $\sigma_1 u_1 {v_1}^\intercal (x_1 - y_1) + \sigma_2 u_2 {v_2}^\intercal (x_2 - y_2)$ instead of $(x_1 - y_1)^2 + (x_2 - y_2)^2$.

Although our classification method uses a prediction (softmax) layer instead of a distance metric, this projection has nonetheless transformed the geometry of the embeddings — giving us additional parameters to represent the desired words best[6].

Figure 2 depicts the difference in the decision surface for both methods. We also note that while we implemented `Word Confusion` as a linear classifier, the method can be easily extended to capture even non-linear relationships between the components in the embeddings by using neural networks in place of the linear projection.

Asymmetry. Human perceived similarity is not symmetric (Tversky, 1977). Yet cosine, like many

[6] Although there are *endless* transformations we can apply to embeddings prior to measuring distances (Mu et al., 2018), the same transformations can also be applied before using `Word Confusion`.

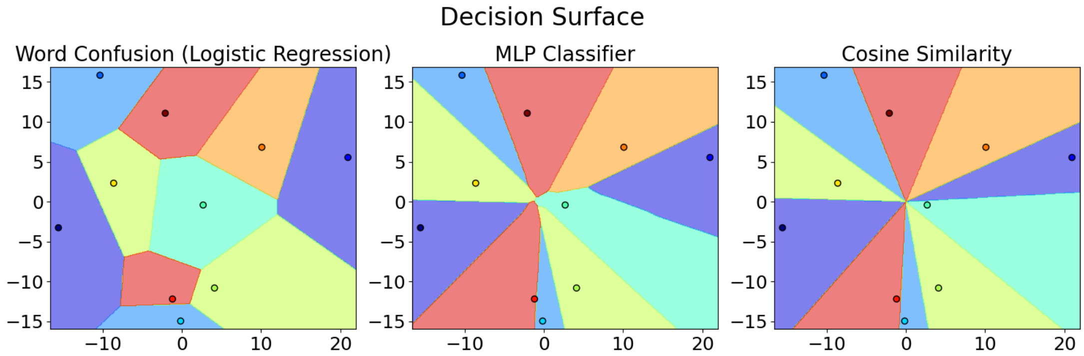


Figure 2: Differences in decision boundaries between `Word Confusion` and cosine similarity. The x and y axes represent two dimensions of an artificially constructed set of data points. Note how cosine similarity's boundaries originate from the origin whereas `Word Confusion`'s are not limited in the same way.

distance functions commonly used to calculate semantic similarity, is symmetric. One of the advantages of using a model's confusion matrix for measuring semantic similarity is that these scores are *asymmetric*; i.e., $p_{ij} \neq p_{ji}$. For example, `Word Confusion` assigns lower probabilities for *animal* being predicted as *cat* than for *cat* being predicted as *animal*. The ability to measure asymmetric semantic similarity opens interesting new directions of understanding semantic similarity which are not possible with cosine.

Domain Adaptability. The fact that `Word Confusion` requires training leads to more flexible similarity measures. Class selection enables measuring the semantic similarity of words relative to just a **subset** of features; we propose that this is particularly useful for practitioners who are interested in computing the similarity of words within a niche domain (we explore this in section 4).

Interpretability. Probabilistic similarity measures have the advantage of being more interpretable for humans than non-probabilistic measures like cosine (Sohangir and Wang, 2017). Using a classifier's confusion matrix gives similarity scores that represent real probabilities. Moreover, since the choice of classifier's classes is an implementation decision, one could choose them based on desired aspects of a word for a task. For example, we could interpret attitudes toward school by asking for the confusion matrix for the word "school" with a sentiment analysis classifier that contains the classes $\{negative, positive\}$, or the classes $\{fun, work\}$.

4 Real-World Data

`Word Confusion` is a new similarity measuring tool that could assist in understanding real-world data and trends. In this section, we focus on two aspects of `Word Confusion` – its ability to serve as a feature extractor and to detect temporal terms in the world.

4.1 `Word Confusion` for Feature Classification

`Word Confusion` can be used to define out-of-domain word classes, i.e. when $w_j \notin W$. Using our earlier example, if the classes of `Word Confusion` are the features $\{positive, negative\}$, given an out-of-domain word like *school*, we can use the confusion matrix to represent the embedding for *school* as a mixture of the classes the model is familiar with, i.e., $\{positive, negative\}$.

Following this intuition, we test whether `Word Confusion` can use features as classes to identify objects' membership to these classes accurately. We used the following tasks:

Sentiment classification using the NRC corpus (Pang et al., 2002; Mohammad et al., 2013). The goal is to classify words according to their sentiment (either positive or negative). The words were manually annotated based on their emotional association (e.g., "trophy" is positive, whereas "flu" is negative).

Grammatical gender classification of nouns (Sahai and Sharma, 2021). We tested `Word Confusion` using two languages – Italian and French. The goal is to classify words according to their grammatical gender per language. For example, "flower" is feminine in French and masculine in Italian.

Domain classification using ConceptNet categories (Dalvi et al., 2022). The goal is to classify words to their correct ConceptNet class. We used two domain pairs: Fashion-Gaming is about clas-

Experiment	`Word Confusion`	Cosine 1	Cosine 2	Cosine 3
Sentiment Classification	**0.83**	0.73	0.73	0.82
Grammatical Gender (Italian)	**0.66**	0.62	0.63	0.51
Grammatical Gender (French)	**0.95**	0.90	0.93	0.79
ConceptNet Domain (Fashion-Gaming)	**0.93**	0.90	0.90	0.90
ConceptNet Domain (Sea-Land Animals)	**0.87**	0.74	0.72	0.78
Average	**0.85**	0.78	0.78	0.76

Table 2: Macro-F1 for `Word Confusion` and cosine similarity across a variety of feature classification tasks. We operationalize cosine similarity in three ways: 1) the distance between the centroids of the seed words and the target words 2) the average distance each of the target word to the centroid of the seed words 3) the average distance of each target word to each seed word (no centroids).

sifying whether a word belongs to the fashion domain or the design domain; in Sea-Land, the goal is to predict if an animal is a sea or land animal.

For each task, we hand-select meaningful words as classes for the classifier and use terms from the lexicon as test embeddings. For example, for sentiment classification we first use the seed words *positive* and *negative* as our classes and collect occurrences from a corpus, extract the embeddings train the concept prober to recognize *positive* and *negative*. Finally, we then use `Word Confusion` to classify all the terms in the NRC lexicon (our target words). We define the label using the class with the highest probability for the word. Details of each experiment are available in in the Appendix C.

Across all three tasks, we find that `Word Confusion` is successful in feature-based classification using a few seed word training examples. Compared to cosine similarity, we achieve a macro-F1 of 83% compared to 73% (see table 2; see C for full results and implementation details).

4.2 What Is A Revolution?

We now offer two pilot studies that look into whether `Word Confusion` could be used to study humanistic or social science concepts. In our first study, we investigate historical changes in the meaning of the French word “révolution”; one of the co-authors of this paper is a French history scholar. Together, we used `Word Confusion` to test a prominent hypothesis of how the meaning of the word and concept of revolution changed (Baker, 1990): that the meaning of “révolution” in the early years of the French Revolution was more associated with *popular* action, but later become identified with *state* actions.

We constructed a set of French words associated with the people ({*peuple*, *populaire*, ...}) and the state ({*conseil*, *gouvernement*, ...}). These seed words were used as classes for our classifier, which we trained on different temporal segments (to capture the temporal change in meaning) extracted from the *Archives Parlementaires*[7], transcripts of parliamentary speeches during a time that contains moments of both emancipation and elite control of political processes. The corpus contains 9,628 speeches and 54,460,150 words from the years 1789-1793. Within this corpus, the term “révolution” appears 2,206 times across 218 speeches, with a contextual basis of 90,138 words.

We color-code the classes (orange as “the people” and blue as “the state”) and project the embeddings down to a 2-dimensional space and visualize the results (figure 3).

We find that, in 1789, the word “révolution” was primarily associated with popular action, the most famous example of which was the storming of the Bastille. In 1790, another definition became common: “révolution” was now also seen as something that the government should lead. Interestingly, we find these instances in the “counter-revolution” cluster indicating that it was primarily when talking about threats to, and enemies of, the revolution, that politicians suggested transferring more power to the state. Jumping forward to 1793, this new governmental meaning had spread back to the word “révolution” itself, when used on its own. Our findings suggest that the goal of repressing counter-revolutionaries is what associated the term “révolution” with governmental action. In other words, once revolutionaries became more concerned about tracking down their enemies, they granted to the government the same kind of extra-legal power that had originally only been the prerogative of the

[7] `https://sul-philologic.stanford.edu/philologic/archparl/`

people in arms.

Our findings are consistent with historians hypothesis that the meaning of revolution in the early years of the French Revolution is most closely aligned with the concept of the people and this gradually shifts as the revolution continues. Furthermore, our model allows us to uncover a potential causal story for this shift in the meaning; that the state sense of révolution first actually started with counter-revolution. This is a novel discovery in our understanding of the French Revolution; future humanistic work should use other methdos to confirm this proposed causal link to counter-revolutionaries.

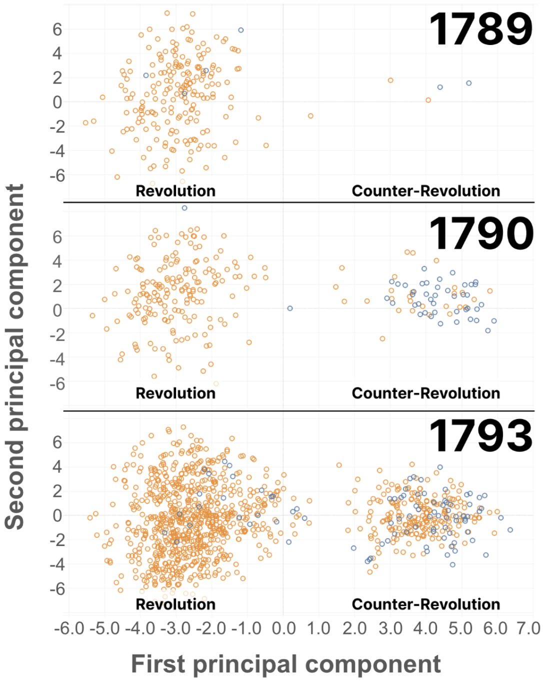


Figure 3: In 1789, the word "revolution" was primarily associated with popular action (represented in orange). In 1790 "revolution" was now also seen as something that the government should lead (represented in blue) found in the "counter-revolution" cluster. In 1793, this new governmental meaning had spread back to the word "revolution" itself.

4.3 Capturing Trends in Inflation

In our second (more speculative) pilot study, we apply `Word Confusion` to a very novel social science domain: representation of financial meaning. Here we test whether we can recover the financial value of goods from their embeddings and use them to predict changes in those values – inflation. We choose inflation since it is easy to quantify and explores a novel domain for this sort of computational meaning.

We used the California Digital Newspaper Collection (CDNC)[8], a newspaper corpus that covers the years 1846-2023. We segmented the data into temporal periods based on trends in the Dow Jones Index (DJI)[9], aggregating intervals that exhibited the same index fluctuation directions. For more details, see Appendix D. At the end of the process, we had 17 different data segments, spanning the years 1915-2009. We then further trained the last layer of a 12-layer BERT model for each temporal segment, to create embeddings that capture a particular historical period, with the goal of capturing the temporal change in the value of money.

To quantify the change in the value of money, we trained `Word Confusion` for every temporal segment of the data. Its goal was to map from the contextual embedding of the " $" token to the (bucketed) monetary value that accompanied that dollar sign. Thus, for each temporal segment, we extract all sentences containing "$", and use the contextual embedding of $ for predicting the bucketed monetary value from the original sentence. For example, if the sentence is "The price of gas increased to $3 per gallon!", we train a linear regression model to correctly map the $ embedding to the bucket that contains 3.[10]

We used all of the temporal `Word Confusion` classifiers to predict the monetary values of items in a typical basket of goods (e.g., egg, milk, gasoline, car, etc)[11]. We then compare these predictions with two measures – the historical Consumer Product Index (CPI) and the Dow Jones Index (DJI)[12]

The correlation between CPI and DJI, is very high (0.966), indicating they capture similar trends. The correlations of `Word Confusion` values with CPI (0.187) and DJI (0.169) are positive and significant but low. This low correlation indicates that inflation prediction is a complicated task, which it looks like we can only very vaguely

[8] https://cdnc.ucr.edu/

[9] https://www.macrotrends.net/1319/dow-jones-100-year-historical-chart

[10] The average correlation coefficient of the trained `Word Confusion` regressors across the different temporal segments is 0.790, indicating a strong correlation between the $ embeddings and their numerical values in context.

[11] To make the analysis as similar to the real CPI as possible, we used the reported products from the website of the U.S. Bureau of labor statistics, keeping only products that were found in all segments (to avoid biasing our results by using products that were not invented in the past).

[12] See Appendix D for other statistics, including correlations with rates of change as well.

approximate using `Word Confusion` (Figure 4). While these second pilot results are inconclusive, they do suggest further study involving domain experts on whether `Word Confusion` could be used to study financial values in text.

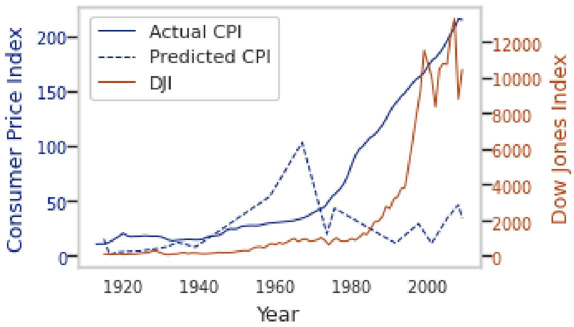


Figure 4: Average CPI, DJI, and `Word Confusion` values between the years 1915-2009. For each temporal segment, the `Word Confusion` values were calculated using the mean predicted value for each item in the basket of goods. We can see that until the 1970s `Word Confusion` values followed the increasing CPI trend, but then dropped. This could be a problem in our method, or could be caused by changes in the training text itself at that period of time, in any case require further investigation that includes domain experts.

5 Related Work on Cultural Change

Both static and contextualized embedding spaces contain semantically meaning dimensions that align with high-level linguistic and cultural features (Bolukbasi et al., 2016; Coenen et al., 2019). These embeddings have enabled a large number of quantitative analyses of temporal shifts in meaning and links to cultural or social scientific variables. For example early on, using static embeddings, Hamilton et al. (2016) measured linguistic drifts in global semantic space as well as cultural shifts in particular local semantic neighborhoods. Garg et al. (2018) demonstrated that changes in word embeddings correlated with demographic and occupation shifts through the 1900s.

Analyzes of contextualized embeddings have identified semantic axes based on pairs of "seed words" or "poles" (Soler and Apidianaki, 2020; Lucy et al., 2022; Grand et al., 2022). Across the temporal dimension, such axes can measure the evolution of gender and class (Kozlowski et al., 2019), internet slang (Keidar et al., 2022), and more (Madani et al., 2023; Lyu et al., 2023; Erk and Apidianaki, 2024).

Lastly, our method has ties with word sense disambiguation (WSD) (Navigli, 2009) and named entity recognition (NER) (Li et al., 2020) and it has been inspired by research and results in these fields. The central idea behind `Word Confusion` of mapping from embeddings to categories are also found in NER and WSD, but instead of focusing on pre-defined concept hierarchies (as for NER) or senses (as for WSD), here we focus on a coherent grouping of words that is interpretable for a given task.

6 Discussion and Conclusion

In this paper, we reframe the task of semantic similarity from one of measuring distances to one of classification confusion. This formulation highlights the context-dependency of similarity judgments, meanwhile avoiding the pitfalls of geometric similarity measures (Evers and Lakens, 2014).

This new framing of semantic similarity in terms of classification confusion introduces new properties that are inspired by cognitive models of similarity (Tversky, 1977) and accounts for the asymmetric nature of semantic similarity, captures different aspects of both similarity and multi-faceted words and ofter a measure that has interpretability benefits

Our proof-of-concept method, `Word Confusion`, demonstrates the practical applicability and effectiveness of this reframing. Empirical results show that it outperforms cosine similarity on standard datasets. For computational social science applications, `Word Confusion` can serve as a way to learn to represent words using target features (e.g., "school" in terms of {*positive*, *negative*}, and can be used to trace the meaning of a word as a function of time (like the $ token and the words "revolution").

The theoretical underpinnings of `Word Confusion` allow it to learn complex word identity boundaries and capture the directional nature of similarity, offering a richer and more flexible framework for understanding word meanings.

While our experiments are preliminary and the space of possible similarity metrics is enormous, we hope this reimagining of semantic similarity will inspire the development of new tools that better capture the multi-faceted and dynamic nature of language, advancing the fields of computational social science and cultural analytics and beyond.

Limitations

Our implementation offers a promising method of where cosine similarity can be replaced by a more sophisticated method that involves self-supervision. However, the boost in performance comes also with some caveats. Because *Word Confusion* is a supervised classifier, it requires an extra training step that simple cosine doesn't require. Furthermore, potential users will need basic understandings of model training and the pitfalls of over-fitting data.

While our experiments were run with a logistic classifier, deeper networks might both help or hurt the performance as it might be more difficult to optimize them. Future work in this area needs to be done.

Another important limitation of our analysis is that our results might be affected by the choice of seed words, since changing seed words can impact the similarities. We explored different sets of seed words without seeing drastic changes in results. However, a robust evaluation of the effect of different seed words should be considered in future work.

Lastly, we are not aware if changing the model used to create the embeddings can degrade the performance; we tested only BERT-Base models.

Ethics Statement

As with all language technologies, there are a number of ethical concerns surrounding their usage and societal impact. It is likely that with this method, the biases known in contextualized embeddings can continue to propagate through downstream tasks, leading to representation or allocation harms. Additionally, the use of large language models for building contextualized embeddings is expensive and requires time and energy resources. To our knowledge, the method we have developed does not exacerbate any of these pre-existing ethical concerns but we recognize our work here also does not mitigate or avoid them.

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Boulder, Colorado. Association for Computational Linguistics.

Keith Michael Baker. 1990. *Inventing the French Revolution: Essays on French Political Culture in the Eighteenth Century*. Ideas in Context. Cambridge University Press.

Paheli Bhattacharya, Kripabandhu Ghosh, Arindam Pal, and Saptarshi Ghosh. 2020. Methods for computing legal document similarity: A comparative study. *arXiv preprint arXiv:2004.12307*.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of artificial intelligence research*, 49:1–47.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Center for Bibliographic Studies and Research, University of California, Riverside. 2024. Courtesy of the california digital newspaper collection. Data retrieved from World Development Indicators, http://cdnc.ucr.edu.

Andy Coenen, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda B. Viégas, and Martin Wattenberg. 2019. Visualizing and measuring the geometry of BERT. *CoRR*, abs/1906.02715.

Fahim Dalvi, Abdul Rafae Khan, Firoj Alam, Nadir Durrani, Jia Xu, and Hassan Sajjad. 2022. Discovering latent concepts learned in BERT. In *International Conference on Learning Representations*.

Katrin Erk and Marianna Apidianaki. 2024. Adjusting interpretable dimensions in embedding space with human judgments. *arXiv preprint arXiv:2404.02619*.

Allyson Ettinger and Tal Linzen. 2016. Evaluating vector space models using human semantic priming results. In *Proceedings of the 1st workshop on evaluating vector-space representations for NLP*, pages 72–77.

Ellen RK Evers and Daniël Lakens. 2014. Revisiting tversky's diagnosticity principle. *Frontiers in Psychology*, 5:57776.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644.

Gabriel Grand, Idan Asher Blank, Francisco Pereira, and Evelina Fedorenko. 2022. Semantic projection recovers rich human knowledge of multiple object features from word embeddings. *Nature human behaviour*, 6(7):975–987.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of the conference on empirical methods in natural language processing. Conference on empirical methods in natural language processing*, volume 2016, page 2116. NIH Public Access.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Nikhil Johri, Daniel Ramage, Daniel McFarland, and Daniel Jurafsky. 2011. A study of academic collaborations in computational linguistics using a latent mixture of authors model. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 124–132, Portland, OR, USA. Association for Computational Linguistics.

Daphna Keidar, Andreas Opedal, Zhijing Jin, and Mrinmaya Sachan. 2022. Slangvolution: A causal analysis of semantic change and frequency dynamics in slang. *arXiv preprint arXiv:2203.04651*.

Austin C Kozlowski, Matt Taddy, and James A Evans. 2019. The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review*, 84(5):905–949.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. 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 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Lucy, Divya Tadimeti, and David Bamman. 2022. Discovering differences in the representation of people using contextualized semantic axes. *arXiv preprint arXiv:2210.12170*.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. 2023. Representation of lexical stylistic features in language models' embedding space. *arXiv preprint arXiv:2305.18657*.

Navid Madani, Rabiraj Bandyopadhyay, Briony Swire-Thompson, Michael Miller Yoder, and Kenneth Joseph. 2023. Measuring social dimensions of self-presentation in social media biographies with an identity-based approach.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. 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 615–621, Minneapolis, Minnesota. Association for Computational Linguistics.

Matej Martinc, Petra Kralj Novak, and Senja Pollak. 2020. Leveraging contextual embeddings for detecting diachronic semantic shift. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4811–4819, Marseille, France. European Language Resources Association.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 321–327, Atlanta, Georgia, USA. Association for Computational Linguistics.

Jiaqi Mu, S. Bhat, and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. *ICLR*, abs/1702.01417.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

B. New, C. Pallier, M. Brysbaert, and L. Ferrand. 2004. Lexique 2 : A new french lexical database. *Behavior Research Methods, Instruments, & Computers*, 36(3):516–524.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics.

Valentina Nicole Pescuma, Chiara Zanini, Davide Crepaldi, and Francesca Franzon. 2021. Form and function: A study on the distribution of the inflectional endings in italian nouns and adjectives. *Frontiers in Psychology*, page 4422.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. 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 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Sebastián A Ríos, Roberto A Silva, and Felipe Aguilera. 2012. A dissimilarity measure for automate moderation in online social networks. In *Proceedings of the 4th International Workshop on Web Intelligence & Communities*, pages 1–9.

Saumya Sahai and Dravyansh Sharma. 2021. Predicting and explaining French grammatical gender. In *Proceedings of the Third Workshop on Computational Typology and Multilingual NLP*, pages 90–96, Online. Association for Computational Linguistics.

Pinky Sitikhu, Kritish Pahi, Pujan Thapa, and Subarna Shakya. 2019. A comparison of semantic similarity methods for maximum human interpretability. In *2019 artificial intelligence for transforming business and society (AITB)*, volume 1, pages 1–4. IEEE.

Sahar Sohangir and Dingding Wang. 2017. Improved sqrt-cosine similarity measurement. *Journal of Big Data*, 4:1–13.

Aina Garí Soler and Marianna Apidianaki. 2020. Bert knows punta cana is not just beautiful, it's gorgeous: Ranking scalar adjectives with contextualised representations.

William Timkey and Marten Van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. *arXiv preprint arXiv:2109.04404*.

Amos Tversky. 1977. Features of similarity. *Psychological review*, 84(4):327.

Luke Vilnis and Andrew McCallum. 2014. Word representations via gaussian embedding. *arXiv preprint arXiv:1412.6623*.

Kelly Zhang and Samuel Bowman. 2018. Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 359–361, Brussels, Belgium. Association for Computational Linguistics.

Kaitlyn Zhou, Kawin Ethayarajh, Dallas Card, and Dan Jurafsky. 2022a. Problems with cosine as a measure of embedding similarity for high frequency words. *arXiv preprint arXiv:2205.05092*.

Kaitlyn Zhou, Kawin Ethayarajh, Dallas Card, and Dan Jurafsky. 2022b. Problems with cosine as a measure of embedding similarity for high frequency words. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 401–423, Dublin, Ireland. Association for Computational Linguistics.

George Kingsley Zipf. 1945. The meaning-frequency relationship of words. *The Journal of general psychology*, 33(2):251–256.

A `Word Confusion` Additional Details

Here, we provide additional details about the experimental set-up of `Word Confusion`.

We used the logistic regression model from the scikit-learn library using a one-vs-rest (OvR) scheme.

Did you try other ways of creating embeddings? We explored alternative methods of creating word embeddings, such as various ways of concatenating layers, but they produced almost identical results.

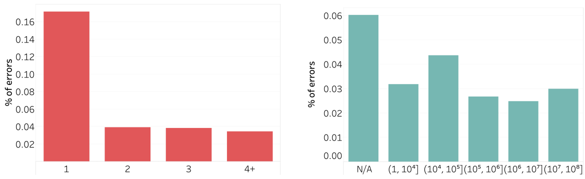
Did you perform any preprocessing? We filtered out short (<20 characters) and long (>512 characters) sentences, and matched keywords on token IDs to ensure punctuation and casing are consistent across examples.

Which hyperparameters did you use? Our task is also trained without any use of hyperparameters or special pre-processing steps to help address the concerns pointed out by Liu et al. (2019); Hewitt and Liang (2019).

How does this differ from BERT's training task and other works? The identity retrieval task differs from the masked LM training task: in masked LM training, the word identity must be predicted from its **surrounding context** rather than the embedding itself. Our task is also related to but different from the "word identity" classifier of Zhang and Bowman (2018) which predicts the identity of a **neighboring** word.

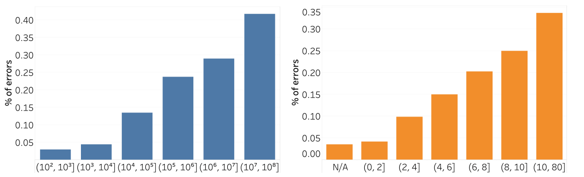
What about OOV words? For the error analysis, we used the embedding of the first subtoken. Throughout the rest of the paper, we average the subtokens following (Pilehvar and Camacho-Collados, 2019) and (Blevins and Zettlemoyer, 2020). Our decision to use the first subtoken in the error analysis section was to investigate the impacts of tokenization and perform analysis on token frequencies of the first subtokens when words were OOV.

In the benchmarking tasks, does your decision to represent a word via the embedding of its first token impact a word's identifiability? We find this is largely not the case. BERT-Base has a ~30,000 token vocabulary, with words that occurred over



(a) Tokens (b) 1st Token Freq

Figure 5: The bar charts above highlight the percentage of errors for words binned by tokens and frequencies of the first subtoken for OOV words. (a) errors by number of tokens (b) errors by frequency of the first token



(a) Frequency (b) Senses

Figure 6: The percentage of errors for words binned by frequency and number of senses.

~10,000 times in its original training data considered in the vocabulary. The word "intermission", is out-of-vocabulary and is tokenized into "inter" and "##mission", and we would use the (extremely ambiguous) first token "inter" to represent "intermission".

Surprisingly, using only the first token to represent an OOV word had little impact on the identifiability of words, suggesting that these embeddings could capture enough context to differentiate themselves from words with identical prefixes. We find that words tokenized into multiple pieces had lower error rates (4%) than words that remained whole (17%) (see figure 5a). In other words, the words "intermission", "interpromotional", "interwar", and "interwoven" are distinguishable from one another even though each is tokenized into "inter" and subsequent tokens and only the first token's embedding is used. That is, the context (namely, the subsequent token "##mission") sufficiently changed the BERT embedding for "inter" to make it identifiable in context. The fact that single tokens words (which are in vocabulary and generally more frequent) performed worse as a group is likely explained by our prior finding that high frequency words have lower performance on this task (see figure 5b).

A.1 Error Analysis

Although `Word Confusion` is relatively accurate, it still makes mistakes, particularly with highly frequent or polysemous words. [13]

[13] Although not critical to this paper, we also include error analysis on the impacts of tokenization and OOV words in Appendix A.

Frequency We find that a word's training data frequency correlates negatively with identifiability. For example, the error rate of words with over 10 million training data occurrences is 42%, compared to an error rate of 3% for rare words with between 100 and 1000 training data occurrences.

Polysemy One explanation for the poor performance of high-frequency words could be the high polysemy of these words (Zipf, 1945). Indeed, `Word Confusion` makes more errors with polysemous words. Very polysemous words (more than 10 senses in WordNet) are 8 times more likely than monosemous words to be misidentified (34% versus 4%, see figure 6b).

Geometric Space Another explanation for lower linear separability of high frequency words is that embeddings of high frequency words are typically more dispersed in geometric space than low frequency words (Zhou et al., 2022b). This would most likely lead to difficulty in identifying them with a simple logistic regression model.

B Details and Full Results from Section 4.1

Implementation Out-of-vocabulary words here are represented as the average of the words' tokens, following (Pilehvar and Camacho-Collados, 2019) and (Blevins and Zettlemoyer, 2020). We experiment with a variety of embedding methods, taking the last layer and taking the first subtoken of out-of-vocabulary words and find comparable results.

Similarity Experiments For cosine, we took 30 samples of each word and we took the average embedding (this is standard practice). For `Word Confusion`, we again took 30 samples and we averaged the vectors of the predicted probabilities before taking the target probability values.

Feature Extraction Experiments Word sampling for target and seed words is done to speed up the computation, we did not find significant differences with different samples (nonetheless, having at least 1000 embeddings to train `Word Confusion` is necessary to get good and stable results).

Models used:

- "bert-base-cased"
- "dbmdz/bert-base-italian-cased"
- "dbmdz/bert-base-french-europeana-cased"

C Seed and Target Words Used

Sentiment Classification

- **Task**: Classifying concepts based on sentiment by using the NRC corpus (Mohammad et al., 2013). Target words: 98 positive and 98 negative words. Seed words: “positive” and “negative”.
- **Corpus**: wikitext-103-v1 from HuggingFace. We remove sentences that are shorter than 15 tokens and longer than 200 tokens.
- **Sampling**: We sample 1000 occurrences of “positive” and 1000 occurrences of “negative”. For each target word, we sample 30 occurrences.

Grammatical Gender in French and Italian
Experiment 1:

- **Task**: Classifying concepts by the grammatical gender of nouns.
- **Corpus**: Latest Italian Wikipedia abstracts from DBPedia. We removed sentences shorter than 20 tokens and longer than 100 tokens.
- **Sampling**: Target words: 140 Italian nouns. Seed words: 59 Italian masculine and feminine adjectives. For each target word, we sample 30 occurrences. For each seed word, we sample 20 occurrences. Seed and target words have been filtered with respect to frequency. Data comes from Flex-IT (Pescuma et al., 2021).

Experiment 2:

- **Task**: Classifying concepts by the grammatical gender of nouns.
- **Corpus**: Latest French Wikipedia abstracts from DBPedia. We removed sentences shorter than 20 tokens and longer than 100 tokens.
- **Sampling**: Target words: 201 French nouns. Seed words: 65 French masculine and feminine adjectives. Seed and target words have been filtered with respect to frequency. Data comes form Lexique383 (New et al., 2004).

BERT Concept Net Classification Land-Sea

- **Task**: Classifying concepts by classes based on the ConceptNet dataset (Dalvi et al., 2022), predicting if an animal is a sea or land animal.
- **Corpus**: wikitext-103-v1 from HuggingFace. We remove sentences that are shorter than 15 tokens and longer than 200 tokens.
- **Sampling**: Target words: 64 land or sea animals. Seed words: category names: “land” and “sea”. We sample 1000 occurrences of each seed word. For each target word, we sample 30 occurrences.

BERT Concept Net Classification Fashion-Gaming

- Task: Classifying concepts by classes based on the ConceptNet dataset (Dalvi et al., 2022), predicting if a concept comes from the fashion domain or the design domain.
- Corpus: wikitext-103-v1 from HuggingFace. We remove sentences that are shorter than 15 tokens and longer than 200 tokens.
- Sampling: Target words: 29 terms related to fashion or gaming. Seed words: category names: “fashion, clothes” and “gaming, games”. We sample 500 occurrences of each seed word. For each target word, we sample 30 occurrences.

D Details and Full Results from Section 4.3

Data Segmentation We segment the temporal data based on the Dow Jones Index trend[14] and aggregate intervals with the same fluctuation directions (see Table 4).

Data Pre-processing We use California Digital Newspaper Collection (Center for Bibliographic Studies and Research, University of California, Riverside, 2024) spanning from 1915 to 2008. The data is pre-processed in the following manner for model continual training:

- Convert all text to lowercase.
- Remove low-quality text corpuses, defined as those where more than 20% of the characters are non-alphanumeric symbols or where more than 20% of words are highly segmented (a single word tokenized into more than two segments), due to poor optical character recognition from scans of historical documents.

[14] https://www.macrotrends.net/1319/dow-jones-100-year-historical-chart

Experiment	Word Confusion	Cosine 1	Cosine 2	Cosine 3
Sentiment	**0.83**	0.73	0.73	0.82
Grammatical Gender (It)	**0.66**	0.62	0.63	0.51
Grammatical Gender (Fr)	**0.95**	0.90	0.93	0.79
ConceptNet (Fashion-Gaming)	**0.93**	0.90	0.90	0.90
ConceptNet (Sea-Land Animals)	**0.87**	0.74	0.72	0.78

Table 3: Full results from Section 4.1. We compare the results of `Word Confusion` to cosine similarity which we operationalize in one of three ways: we measure cosine similarity in one of three ways 1) the distance between the centroids of the seed words and the target words 2) the average distance each of the target word to the centroid of the seed words 3) the average distance of each target word to each seed word (no centroids)

Year	DJI Avg. Annual Change
1915	81.49%
1916-1917	-12.95%
1918-1919	20.48%
1921-1928	20.48%
1929-1932	-31.67%
1933-1936	30.02%
1937-1941	-7.16%
1956-1961	9.97%
1962-1972	3.86%
1973-1974	-22.08%
1975-1976	12.35%
1988-1995	13.53%
1996-1999	22.49%
2000-2002	-10.01%
2003-2007	11.04%
2008	-33.84%
2009	18.82%

Table 4: Years aggregated by DJI fluctuation directions

- The dataset of each training segment has 10,240 training documents, 1280 test documents and 1280 validation documents, each containing an average of 350 tokens.

Continual Training We fine-tune the last layer of the 12-layer bert-base-uncased model, which comprises 7,087,872 trainable parameters. We use a learning rate of 2×10^{-5} and a weight decay of 0.01. Each model takes 3 hours to fine-tune with Google Cloud T4 GPUs.[15]

Training `Word Confusion` We extract 2,000 occurrences of the "$" token from each segment. Each token is part of a 128-character window and must be followed by a numeric value. We get the contextualized embedding of the tokens using the fine-tuned models and bucketize the 2000 numeric values into 60 buckets to reduce noise in the data. We then train a linear regression for each time segment.

Calculating CPI To calculate the Consumer Price Index (CPI), we construct a basket of goods consisting of the following items: {"car", "rent", "hat", "wine", "jewelry", "shirt", "chicken", "milk", "furniture", "egg", "shoe", "pork", "gasoline", "beef", "coffee", "bus"}. We identify occurrences of the "$" token that are followed by a numeric value and keep those where terms from our basket of goods appear within a 20-word window. The numeric values are then masked, and the trained `Word Confusion` classifier is used to predict the value associated with each "$" token.

Models used:

- “bert-base-uncased"

[15] https://cloud.google.com/compute/docs/gpus#t4-gpus

Rate of change in CPI, DJI, and `Word Confusion` values: Rate of change in `Word Confusion` values compared with the rate of change in CPI and DJI values (the mean annual change in values per temporal segment). The correlation between the change in CPI and DJI values is almost zero (-.006), suggesting they capture quite different trends. The correlation of CPI change and `Word Confusion` change is negative (-0.226), and the correlation between the changes in DJI and `Word Confusion` values is positive and significant (0.387).