Eurographics Conference on Visualization (EuroVis) 2018
J. Heer, H. Leitte, and T. Ropinski
(Guest Editors)

Volume 37 (2018), Number 3

Interactive Analysis of Word Vector Embeddings

F. Heimerl[1] and M. Gleicher[1]

[1]Department of Computer Sciences, University of Wisconsin–Madison, USA

Abstract

Word vector embeddings are an emerging tool for natural language processing. They have proven beneficial for a wide variety of language processing tasks. Their utility stems from the ability to encode word relationships within the vector space. Applications range from components in natural language processing systems to tools for linguistic analysis in the study of language and literature. In many of these applications, interpreting embeddings and understanding the encoded grammatical and semantic relations between words is useful, but challenging. Visualization can aid in such interpretation of embeddings. In this paper, we examine the role for visualization in working with word vector embeddings. We provide a literature survey to catalogue the range of tasks where the embeddings are employed across a broad range of applications. Based on this survey, we identify key tasks and their characteristics. Then, we present visual interactive designs that address many of these tasks. The designs integrate into an exploration and analysis environment for embeddings. Finally, we provide example use cases for them and discuss domain user feedback.

CCS Concepts

•***Visualization*** $\rightarrow$ *Information Visualization; Visual Analytics;* •***Artificial Intelligence*** $\rightarrow$ *Natural Language Processing;*

1. Introduction

Word embeddings are mathematical models that encode word relations within a vector space. They are created by an unsupervised training process based on co-occurrence information between words in a large corpus. The encoded relations include semantic and syntactic properties of words. For example, word embeddings have been shown to reveal semantic analogies [MCCD13] and groups of semantically related words [PSM14].

Due to their ability to capture word meaning, embeddings are valuable in many diverse applications. They are particularly popular in natural language processing (NLP) applications because of their potential to significantly improve accuracy of language processing methods. Examples include text classification [KSKW15], sentiment analysis [YWLZ17], and natural language parsing [SBM*13]. Embeddings have also sparked the interest of linguists and researchers from the humanities. For them, word vector embeddings can provide valuable insights into the use and structure of language. Examples include etymological studies of word meanings [HLJ16], and assembling dictionaries [FCB16].

These diverse scenarios come with a variety of challenges to understand and compare embeddings. They range from learning how to interpret similarity in the vector space, to understanding the influence of source corpora on the resulting embeddings. For example, neighborhood relations are often used to probe embeddings for specific information. But, there are many reasons words may be close in an embedding. They may have close semantic meanings, or similar syntactic roles within sentences in the source data set. In addition, embedding algorithms are non-deterministic and depend on critical input parameters, including the dimensionality of the resulting word vector space. This can lead to quite different embeddings even with identical input data sets, making their interpretation [LG14] and evaluation [LHK*16, BGH*17] challenging.

Interactive visual interfaces are effective for the task of analyzing word vectors embeddings, because (1) the problems to be solved are inherently human-centric and finding solutions involves enabling expert users to gain a deeper understanding of word embedding spaces, for which visualization is a primary tool, and (2) while effective visual encodings can make interesting features readily available, human linguistic and domain knowledge is necessary to drive the process and ultimately define relevant and non-relevant artifacts of the data and are a strong motivation for interactive visualization. However, such exploratory processes require tightly integrated feedback loops that let users navigate, filter, and drill down onto aspects relevant to them and their specific analysis goals.

While visualization has been used to analyze word embeddings in the NLP and digital humanities (DH) literature, it is often based on standard dimensionality reduction techniques. Such tools convey a rough impression of similarities but fail to serve a broader range of tasks. Recent work [LBT*17] has identified specific tasks in analyzing word vector embeddings and shown that these tasks can be addressed with visualization tools, but focuses on one particular task.

In this paper, we bring a broader, task-based design process to the design of visualization tools for word vector embeddings. We aim to derive a range of tasks, including investigating neighborhoods and reconstructed co-occurrences, and aligning word vectors based on concept axes. We derive these tasks based on domain literature and collect relevant characteristics of word vector spaces for evaluating, testing, and using them. We then introduce visual interactive designs to help gain insights into embeddings, and show their capabilities based on use cases. Finally, we briefly discuss domain user feedback. We see our contributions as:

- A review of domain literature and a collection of domain specific tasks that word embeddings are used for and evaluated with;
- Identification of the characteristics of word embeddings relevant for the domain tasks as the targets of practical analysis tasks;
- Visual interactive designs that support several of those tasks.

2. Background and Related Work

Word vector embeddings place words as points in a vector space. The positions are designed to encode meaning, based on the concept of distributional semantics—the assumption that words that appear in similar contexts are close in meaning [RG65]. Meanings are captured by statistically quantifying the contexts of a word across a large body of text. Word embeddings can encode semantic similarity by placing similar words close in the space [PSM14] and semantic analogies, e.g., *king* is to *queen* as *man* is to *woman* [MCCD13] as arithmetic operations.

Building a word vector embedding requires a text data set, large and broad enough to contain ample information about each single word. After tokenization, the corpus is processed and converted into a co-occurrence matrix containing the frequency of two words occurring together in a window of a certain size. One way of viewing word embedding algorithms is as a factorization of the co-occurrence matrix. This can either be done directly based on the frequencies [PSM14, MSC*13], or after converting them to another association measure, e.g., pointwise mutual information [ALL*16]. The embedding algorithms factor the co-occurrence matrix into two lower dimensional ones that each contain one vector for each word. The number of dimensions is a free parameter (typical values are 50-300). The two vectors for each word are called word vector and context vector, respectively. Context vectors are often discarded after building the model, but can be useful during analysis (see Section 4).

While we focus on embeddings of single words, embedding methods have been extended and improved in various ways. Examples are embedding entire sentences [ALM17], and methods that focus on more than one language [ZSCM13], a valuable resource for machine translation. In addition, they have proven useful for linguistic studies [HLJ16], and for creating dictionaries [FCB16].

2.1. Model Visualization

Helping stakeholders gain a better understanding of data-driven models is an important and broad challenge [Gle16]. Visualization research has approached it from various angles. Several approaches have inspired and influenced our work.

Readily interpretable models, such as decision trees, allow methods for direct manipulation [vdEvW11], and model selection [MLMP17]. However, most embeddings are too complex or abstract to follow each step of the modeling process, calling for different strategies. Notable approaches for classifiers are EnsembleMatrix [TLKT], that allows users to combine multiple single models to improve quality, while Heimerl et al. [HKBE12] let users interact with and modify a model over multiple iterations. These approaches are a motivation to provide users insight into word embedding models. However, while direct manipulation techniques are conceivable for word embeddings in the future, they depend on effective methods that let users gauge their quality, which we focus on.

Other approaches create simpler models to help users interpret complex ones. Dimensionality reduction, such as t-SNE [MH08], creates low-dimensional embeddings of high-dimensional data sets and retains local similarities to convey neighborhood structure. It is popular to view local neighborhoods in embeddings, but we found it to be not helpful for the tasks we aim to support (see §4.1.1 for details). Explainers [Gle13] and Interaxis [KCPE16] let users define semantic axes for projection, an idea we will use in §4.3.

Some vector techniques (e.g., Word2Vec [MSC*13]) use neural networks to construct the embeddings. Visualization approaches can help in understanding such networks. For example by grouping and visualizing nodes to understand their role [LSL*17], analyzing the training process [LSC*17], creating abstractions for complicated networks [WSW*17], or conveying local changes in a network [SGPR17]. However, analyzing these network structures does not help with understanding resulting word embeddings.

Most closely related to ours, there are previous projects that aim at visualizing word embeddings. Rong and Adar [RA16] focus on the training process, but do not support tasks to analyze the resulting embedding. Liu et al's. [LBT*17] focus on a subset of tasks that aim at analyzing analogy relationships in embeddings. Smilkov et al. [STN*16] aim to support neighborhood analysis tasks using dimensionality reduction to create 2D and 3D layouts of embeddings.

2.2. Text Visualization

Topic models summarize text corpora by embedding documents in a vector space. Visualization approaches for them inspire our work on word embeddings. Similar to us, Serendip [AKV*14] provides multiple views on different aspects of a model. Alexander and Gleicher [AG16] focus on comparing topic models. We also serve comparison tasks with our approach. TopicPanorama's [WLL*16] hierarchical design has inspired our neighborhood view (§4.1), which provides an overview over change patterns and lets users explore details on demand. To support search tasks, Choo et al. [CLRP13] use a 2D layout of documents based on a topic model, inspiring our 2D plots in §4.3.

Visual interactive approaches to text data have a long tradition in visualization research. Jigsaw [SGL08] visualizes entity connections extracted from large text collections. Heimerl et al. [HJH*16] design a scalable solution for exploration at various levels of detail based on a variation of the excentric labeling technique [FP99, BRL]. Scattertext [Kes17] tackles the problem of visually comparing two corpora based on word frequencies. Collins et al. [CCP09] visualize hierarchical semantic relations between prominent words in a text. Those approaches are designed to analyze text data directly, while our focus lies on understanding word embeddings. One of our tasks aims at co-occurrence patterns, which have been addressed before. Wattenberg and Viégas [WV08] visualize co-occurrence sequences using an interactive tree design. PhraseNets [vHWV09] display more complex phrase patterns as a graph. While those designs show co-occurrences as longer sequences of words, word embeddings encode pairwise statistics, requiring a different solution.

Recently, visual text analysis methods that rely on word embedding techniques have been proposed. ConceptVector [PKL*17] focuses on text analysis and retrieval based on user-defined concepts that are built by identifying related words within an embedding. Berger et al. [BMS17] embed citations from scientific literature together with words, providing information about the reasons for a citation. Both approaches use embeddings to analyze text data, while we aim to provide insight into the embeddings themselves.

3. Task Analysis

We seek to help researchers and practitioners better use word vector embeddings through improved tools. To best do this, we begin by identifying the tasks users attempt [SMM12, Mun14]. We base our analysis on a survey of domain literature. Specifically, we use a structured literature analysis to understand domain specific problems and collect tasks. For this, we compiled a representative collection of publications from which we extract relevant tasks based on word vector properties important to experts.

We rely on literature from the NLP community and related fields. In addition, we include papers from linguistics and the DH that use word embeddings. For this, we compile an initial set of articles published by the Association for Computational Linguistics (ACL) [†], the primary publication source for word embedding literature. We did this by querying for all publications that have *word embedding* in their title. We then skimmed them for relevant information about use and evaluation of word embeddings. In addition, we followed backward and forward citation links to obtain additional relevant material. Overall, we surveyed 111 papers from a diverse set of communities, including data mining and human computer interaction. A table of the results, with labels and their descriptions, can be found in the supplemental material. It includes all 111 papers reviewed and lists additional citations mostly from the visualization community.

Our goal is to understand domain tasks for word embeddings. For this reason, we do not discuss visualization approaches in this section, and only focus on tasks and task analyses. In case the reviewed papers propose visualizations, we compare them to ours either in §2 or discuss them as design alternative in §4.

[†] https://aclweb.org/anthology/

3.1. Linguistic Tasks for Word Embeddings

Word embeddings are either used as a tool to learn more about language, or as part of an NLP pipeline. When using word embeddings as part of an NLP application, the key human task is to evaluate the embeddings. Evaluating embeddings is challenging as there is no ground truth answer, which has lead to a variety of evaluation approaches. Evaluation falls into two categories [SLMJ15]: *extrinsic* and *intrinsic*. *Extrinsic* evaluation, gauges the quality of a word embedding based on its application scenario in language processing [NAM16, YS16]. For example, when an embedding is used as part of a machine translation pipeline, its quality is measured through the positive effect on overall translation results. Tasks such as machine translation do not target embeddings, but specific scenarios for NLP applications in which embeddings are just one part of the pipeline. Therefore, we do not consider extrinsic evaluation as a source of embedding tasks.

Intrinsic evaluation is a major source of embedding tasks for users. This type of evaluation seeks to measure the usefulness of the encoded relations by comparing them to known word relations based on lexical resources. Examples include semantic similarity of words or analogy relations [LG14]. Intrinsic evaluation is usually based on human-coded data sets in which these relations are encoded explicitly. In addition, examples of such relations are used in the literature to give readers a qualitative notion of embeddings, e.g., by listing the nearest neighbors of selected words [YWL*16]. Because they are based on human linguistic knowledge and provide direct insight into encoded relations, they are highly relevant for identifying important characteristics of embeddings.

In addition, we include practical analysis tasks from other domains, such as the DH in our task analysis. Because intrinsic evaluation focuses on linguistic relations, tasks from both domains overlap to a certain extent. For example, an intrinsic evaluation might look at synonymy between words. Similarly, DH researchers might be interested in exploring synonymy relations within the specific data set the embedding was trained on. Tools for discovering and analyzing synonymy can help both groups.

linguistic tasks	**characteristics**	**examples**
rank word pairs	similarity	[BDK14, PSM14]
compare concepts	average, similarity	[RBS17, SLMJ15]
find analogies	offset, similarity	[SLMJ15, LG14]
view neighbors	similarity	[HLJ16, YWL*16]
select synonyms	similarity	[BDK14, FDJ*14]
project based on concepts	concept axis	[BCZ*16, FRMW17]
predict contexts	co-oc. probability	[SN16, LJW*15]

Table 1: *Linguistic tasks for which word vector embeddings are used. The tasks are based on domain literature, with examples listed for each one.*

The literature provides a range of analysis and evaluation procedures. We have extracted a list and abstracted them into a set of linguistic tasks, listed in Table 1. The second column of the table lists vector operations used to extract the information encoded within the embeddings, while the third column lists a small number of example publications for each of the linguistic tasks as an illustration. The tasks comprise:

- *rank word pairs*

The task is to rank word pairs in order of descending similarity. Evaluation data sets contain pairs ordered by human annotators. To create an ordering based on embeddings, cosine similarity is mostly used to quantify similarity and order word pairs. The evaluation metric is the correlation between those results and the human-generated ones, e.g., [PSM14].

- *compare concepts*
  We use the term concept to denote an arithmetic structure obtained by averaging a set of word vectors to encode a common concept. For example, a concept can represent the selectional preferences of a verb [BDK14], or phrases and sentences [RBS17]. The goal is to evaluate the similarity between a concept and a word (or another concept), e.g., to find nouns a verb prefers as objects. Again, cosine similarity is mostly used to quantify similarity.
- *find analogies*
  Evaluations based on word analogies use lexicons that contain tuples of word pairs where both pairs have the same relation. Those relations can be of different grammatical types. Examples are *woman* is to *man* as *queen* is to *king*, and *flying* is to *flew* as *helping* is to *helped*. Evaluations based on such analogies are done by guessing one of the components of an analogy. For this, similar vector offsets between both elements of a pair (e.g., $v(woman) - v(man) \simeq v(queen) - v(king)$) are assumed [MCCD13]. The missing component can then be identified through a nearest neighbor search (e.g., $v(queen)$ as the nearest neighbor of $v(king) + v(woman) - v(man)$).
- *view neighbors*
  Viewing and comparing the neighbors of single word vectors is often used in the literature as a form of qualitative evaluation of the word space, for example in [YWL*16]. In addition, linguistic studies into changes of word meaning over time have been based on comparing nearest neighbors of a word across multiple vector embeddings trained on corpora from different eras [HLJ16].
- *select synonyms*
  For a given word, the goal is to select a synonym from a list of candidate expressions. An example for this is selecting a synonym for *rug* from *sofa*, *ottoman*, *carpet*, and *hallway* [FDJ*14]. This particular evaluation strategy is almost identical to ranking word similarities, but only focuses on semantic relations. Again, it is typically solved by selecting the candidate expression that is closest to the vector of the given word based on cosine similarity.
- *project words on concept axes*
  Projections of a selected set of word vectors onto an axis that represents a certain concept is used in the literature as a way to qualitatively explore word embeddings. This is typically done by using the difference between two words that represent two opposite concepts (e.g., *happy* and *sad*). The set of word vectors is then projected onto this axis through a linear projection. Depending on the encoded information within the embedding, their position can be interpreted as a word's affinity to each of the opposing concepts. One usage example of such projections is identifying gender biases encoded in word embeddings [BCZ*16].
- *predict contexts*
  The most common word embedding algorithms encode co-occurrence frequencies or probabilities between words in the training corpus. Co-occurrence probabilities have been used to evaluate word embeddings based on sentence completion data sets that contain sentences that miss one word. Word embeddings can be used to predict the correct word by averaging the predictions of all words in the missing word's context [LJW*15].

3.2. Characteristics and Practical Tasks

Table 2 lists four characteristics of word vector embeddings (rows 1 through 4) that are pertinent to the linguistic tasks. We link each to two practical tasks that help analyze these characteristics in embeddings, and support the linguistic tasks. There is thus two different sets of tasks mentioned in this section, the linguistic tasks that word embeddings are used for, and the practical tasks that help users understand word embeddings. The latter are derived to help users analyze embeddings for specific linguistic tasks. Based on the terminology by Schulz et al. [SNHS13], each of the characteristics is linked to a *target*, which is an aspect of a data item that a task focuses on. Following [Mun14], we define tasks as actions (verbs) and targets. Table 2 provides tasks for both single targets and comparison (multiple targets). Comparison tasks can either focus on targets from different embeddings, or on multiple targets from a single embedding.

characteristic	**single target**	**multiple targets**
similarity	(1) inspect local neighborhood	(2) compare local neighborhoods
average, offset	(3) inspect arithmetic structure	(4) compare arithmetic results
co-oc. probability	(5) analyze encoded probabilities	(6) compare probabilities
concept axis	(7) analyze vector relations	(8) compare vector relations
multiple	(9) discover interesting aspects	(10) compare interesting aspects

Table 2: *Ten tasks that help users understand and compare word vector embeddings. The tasks cover different levels of granularity, from exploratory ones to ones targeted at specific characteristics.*

The first characteristic listed is similarity. Similarity among vectors, most commonly quantified by the cosine between them, is the most common way of evaluating and comparing linguistic relations between words. Here, inspecting the structure of neighborhoods can help understand what similarity means for a certain embedding. For example, a user might be interested in what type of relations the nearest neighbors have to a specific word, and whether they tend to be more of a syntactic or a semantic nature. In addition, comparing the similarities and differences in the neighborhood of two different words helps understand aspects in which they differ. Similarly, comparing neighborhoods for words across embeddings helps to understand what discriminates different embeddings.

The next characteristic are arithmetic structures that represent meaning in the embedding space. They are derived from basic word vectors through arithmetic operations. In the literature we mainly found two different operations to create new vectors: averaging words to encode a joint concept, and vector differences (offsets) to represent specific relationships between words. Here, analyzing and comparing such structures the same way as regular word vectors, i.e., based on their local neighborhoods, helps understand the concepts they represent. In addition, finding interesting relations within the space by identifying meaningful vector offsets is another challenge that can help to uncover hidden structure within an embedding. Again, comparing those relations between embeddings can provide valuable insights into differences in the encoded relations .

Analyzing word co-occurrence probabilities helps uncover distributional information encoded within an embedding model from the source corpus. While this information can be used to gauge the likelihood of a sentence, either to complete it or to find errors, it can also help to learn about the influence of the underlying corpus on an embedding. For example, a user might be interested in learning the co-occurrence differences between two words in two embeddings trained on different corpora to understand why their similarity differs in both models.

As described above, laying out words according to axes that distinguish between two opposing concepts reveals substructures of the embedding. The concepts can be of different types, depending on a user's analysis goal. Analyzing and exploring such a mapping within one embedding helps uncover latent structures in the vector space. Comparing them between different embeddings can help gauge the influence of corpora, algorithms, and parameters on capturing such structures.

The two final tasks in Table 2 target high-level characteristics [SNHS13] of an embedding, and are not directly derived from the linguistic tasks in §3.1. They are included because providing global overviews is an important first step in a large variety of analysis scenarios [Shn96]. However, these tasks are still motivated by our findings from the review. In a significant portion of the literature, exploratory methods are used to assess the results of embedding algorithms. They help to initially gauge an embeddings quality and identify potentially interesting properties for closer scrutiny. The importance of exploratory methods has therefore been acknowledged in the literature [GD16]. Moreover, word embedding users from the DH community rely on exploration of embeddings to find interesting connections, as numerous blog posts form this community show[‡]. Exploration, for example, can start with some text of interest to a researcher, such as a specific novel, whose language is to be analyzed based on embeddings. While these methods are seen distinct from evaluation with annotated resources, those resources can be used in an exploratory environment to help guide users to interesting local phenomena, for example, clusters of words with a particularly high error rate.

4. Interactive Exploration of Embeddings

Most, if not all, of the identified tasks can benefit from appropriately designed visualizations. However, existing ones show the global structure of the space (e.g., a t-SNE embedding) or focus on analogies (e.g, [LBT*17]) and therefore do not address most of these tasks directly. To complement these existing visualizations, we consider designs that address three other key task groups: exploring neighborhoods, reconstructed co-occurrences, and mapping words to concepts. In contrast to previous approaches, we thereby focus on locally confined relations among a limited number of vectors.

[‡] two examples are: http://bookworm.benschmidt.org/posts/2015-10-25-Word-Embeddings.html and http://ryanheuser.org/word-vectors-1

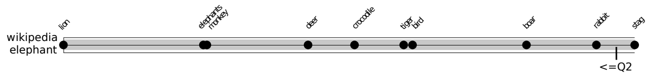

Figure 1: *The ten nearest neighbors of* elephant. *Distance on the axis encodes distances from the selected word in the embedding space. To get a sense of the density of the neighborhood compared to others in the embedding, hovering the gray padding shows a tool tip with the quartile in which the size of the neighborhood falls ($<=$ Q2 means that the shown neighborhood is in the second quartile).*

For the examples and screen shots in this section, we use embeddings from two different corpora, both using GloVe [§] (window size: 15, dimensions: 50). The first one is trained on the EEBO-TCP [¶], a corpus that contains a collection of historic English texts from before 1700. The second one is trained on the English wikipedia.

4.1. Local Neighborhoods

4.1.1. Inspecting Single Neighborhoods

Task (1) in Table 2 is inspecting local neighborhoods. For this, users should be able to view the closest neighbors of a vector, and get a sense of the distances relative to others. In addition, having a means to judge the relative density of the neighborhood within the embedding helps gauging the significance of the neighbor relation. The proposed design to inspect neighbors also serve **task (3)**, inspecting arithmetic structures. One of their important aspects is their local neighborhood [MCCD13].

Our design is shown in Figure 1. It depicts the nearest neighbors of a chosen word as points along an axis. Our examples use the cosine distance as this is the most common choice in the literature. Compared to an ordered list, in this design, the distances between the selected word and each of the neighbors reflect those in the embedding. An issue with this design is that while distances to the chosen word are represented accurately, those between neighbors are not. An obvious alternative would be a 2D scatter plot of vectors in a local neighborhood. According to our experience, the popular 2D projection methods (including PCA, MDS, and t-SNE) are able to retain some of the pairwise differences, with MDS producing the most interpretable results for a small local set of word vectors (10-20). However, while the resulting 2D displays convey additional information about neighborhood structure, it remains unclear how much of it is noise that is captured in the process as compared to relevant semantic structure. This makes interpretation, and in particular comparison across embeddings challenging. We were thus unable to find a concise visual encoding, and see the 1D plots as a good trade off between providing insight into local distances and facilitating comparison (see §4.1.2).

One challenge with single neighborhoods is providing context

[§] https://nlp.stanford.edu/projects/glove/

[¶] http://www.textcreationpartnership.org/tcp-eebo/

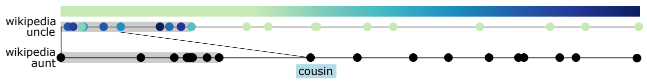

Figure 2: *Comparing the differences in neighborhoods between* uncle *and* aunt. *The colors on the first axis encode positions in the second axis. We can see that there is no overlap between the neighborhoods, but that one nearest neighbor of uncle is closer than the others to* aunt *(it is selected and linked across the axes). Hovering it reveals that it is the word* cousin *(shown by the tooltip).*

about its density relative to others in the embedding. This information helps to gauge how meaningful the neighborhood relationship is. If the nearest neighbors are comparatively far from the selected word, the fact that they are its nearest neighbors might not mean that they share any relevant properties. To solve this, each neighborhood has a grey padding behind the axis that reveals a tool tip with the quartile of neighborhood sizes in the embedding it falls into (see Figure 1). As described later in detail, the grey padding is also important when comparing multiple neighborhoods and their relative sizes.

Users can choose an interesting word whose nearest neighbors they want to inspect, and the size of the neighborhood to display. While qualitative analysis of nearest neighbors in the literature typically uses a very small number of neighbors (3-5), we found that 10 is a good number for many scenarios. Figure 1 shows the 10 nearest neighbors of *elephant* in the wikipedia embedding as an example. We can see that all the nearest neighbors are semantically related to *elephant* (it is apparent that they all share the same hypernym *animal*). In addition, we can see the plural of *elephant* as its second neighbor. This gives us first insights about how to interpret similarity in this embedding.

4.1.2. Comparing Multiple Neighborhoods

Task (2) from Table 2 is comparing multiple neighborhoods. In addition to comparing the set of neighbors, differences in their rank, and the absolute and relative distances to the chosen word are relevant. Comparing neighborhoods of word vectors helps to gauge aspects in which words are similar or differ within the embedding (see example below). Again, similar to the previous designs, this one also serves **task (3)**, as comparing neighborhoods provides useful insights into similarities and differences of arithmetic structures. In particular, comparing the neighborhood of arithmetic structures to those of word vectors, can provide valuable insights (see example below).

We extend the previous design using ideas from *buddy plots* [AG16], and stack multiple axes, each of which depicts the neighborhood of one word in an embedding (see Figure 2). The distance from the selected word in the selected embedding for each axis is again encoded as position. In addition, the position of each word on the axis following the current one is encoded using a color ramp. For reference, the color ramp used is shown on top of the plot. Words on the last axis are colored black. Since the sets of nearest neighbors likely contain different words across the axes, they show the union set of all neighborhoods to keep them comparable. The *n* nearest neighbors (*n* chosen by the user) are marked by a gray padding on the left side of each axis. When neighborhoods across different embeddings are compared, some of the embeddings may not contain all of the words. Their circles are consequently omitted on the respective axis, and are marked black on the preceding one. In order to explore neighborhoods and highlight differences, users can select single words, which are then connected through a line (*cousin* in Figure 2).

As the design facilitates comparison between neighboring axes, their order is relevant to allow for pairwise comparison if a natural ordering exists (see §6 for an example). Depending on the scenario, users may want to change the reference axis for the color coding to quickly compare different axis pairs. To support this, users can select an axis, which then becomes the reference for all others. The other axes then encodes the distances on the selected one as color, allowing for a pairwise comparison between one axis with all others. The selected axis encodes the same distances twice, resulting in a perfectly smooth gradient (see second axis in Figure 3).

The proposed design lets users easily identify differences from one axis to the next one: the smoother each gradient is for an axis, the less changes there are in the order of neighbors compared to the following axis. While color is not the most effective visual variable to encode absolute positions, changes in relative position can be easily discerned as breaks in the color gradient. The most prominent design alternative is one based on parallel coordinates, linking points on neighboring axes through lines. However, while the number of edge crossings would give an impression of the degree of changes between axes, such a solution is very prone to clutter, hindering comparison, and, in particular the identification of single words that change position. In addition, changing the reference axis for comparisons would not be possible without reordering the view for each pairwise comparison, which would demand lots of interactions from users. Another design option is to directly show the word of each neighbor along the axis. Parallel Tag Clouds [CVW09] is an approach that facilitates comparison among multiple corpora by laying out word clouds along an axis for each of the corpora and marking overlaps. However, in our case, the number of overlaps tends to be very high, which would lead to constant repetition of the same words. In addition, showing terms constantly introduces lots of clutter that hinders the identification of change patterns across multiple axes. For this reason, we have decided to omit words by default, and offer advanced tool tips that show the words for single axes on demand (as illustrated in Figure 3).

Figure 2 compares ten nearest neighbors of *uncle* to *aunt*. Based on their color, we can see that all of them are further away from *aunt* than they are from uncle. Inspecting both neighborhoods reveals that they mostly contain female family relations for *aunt* and male ones for *uncle*. The word from the neighborhood of *uncle* that is closest to *aunt* (the selected and linked one in Figure 2) is the gender neutral *cousin*.

We now compare two offsets to capture the difference in meaning between *aunt* and *uncle*, namely the offset between *man* and *woman* and that between *brother* and *sister*. Figure 3 compares the neighborhoods of both offsets added to *aunt*. In addition, we compare both of the neighborhoods to *uncle*, to get a sense of their similarities and differences. Both structures have *uncle* as their nearest neighbor (the two black circles). While the bottom structure seems

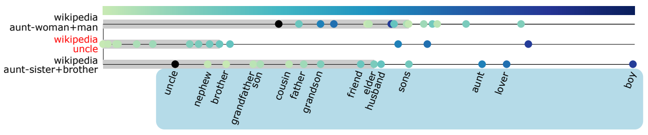

Figure 3: *Comparing the offset between* man *and* woman *and that between* brother *and* sister *when added to* aunt *to the word* uncle. *The axis for* uncle *(red) is selected as the reference for the others. This reveals an interesting pattern. Both resulting vectors have* uncle *as their nearest neighbor (the two black circles). While the bottom structure seems to have a similar neighborhood to* uncle *(most of the words are male relatives, including* brother, nephew, *and* grandfather*), this is not true for the vector on top. The words are shown as tooltips when hovering them with the mouse, either individually, or, as shown for the bottom axis, for an entire axis.*

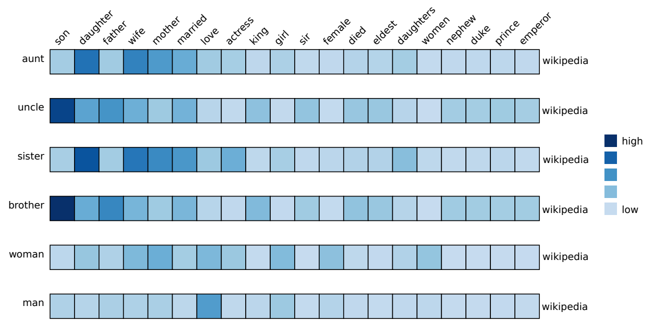

Figure 4: *Comparing the reconstructed co-occurrences between a set of words. The rows of the matrix show selected words, the columns show the reconstructed co-occurrences (the set of reconstructed co-occurrences shown as columns in the matrix is selected based on their high variance between the selected words). Color encodes the reconstructed strength of co-occurrence. Words that denote family relations and nobility titles have higher reconstructed co-occurrence strengths with the first four terms.*

to have a rather similar neighborhood to *uncle* (most of the words are male relatives), this is not true for the one on top. Most of its nearest neighbors are disjunct with those of *uncle*. Inspecting them more closely reveals that they are either words denoting female relatives (e.g., *aunt*), or more general terms (e.g., *boy*). Based on this, we can hypothesize that while $vec(man) - vec(woman)$ captures the male to female relation in general within the vector space, $vec(brother) - vec(sister)$ more accurately captures the more specific female relative to male relation.

4.2. Approximate Co-occurrence Patterns

Tasks (7) and (8) target co-occurrences encoded within a model. Embedding algorithms are constructed in a way that some measure of co-occurrence is retrievable from the model for a word pair after training. We use this to visualize approximate co-occurrences encoded in the models. This is helpful to interpret and understand results when embeddings are used for context prediction (see §3). Analyzing the encoded co-occurrences also provides useful insights into the underlying corpus. While directly showing values from the training corpus is intractable, we use these co-occurrence estimates instead. In addition, it can help to debug embeddings (see §6 for an example).

To support the task of analyzing and comparing reconstructed co-occurrences, we use a matrix-based design (Figure 4). Each row corresponds to a word, with their co-occurrences along the columns of the matrix. Co-occurrence strength is encoded using color, with a legend to the right of the matrix. We have explored alternative designs, most notably rows of bar charts, but found that color encodings allow for easier comparisons, and finding patterns. Another notable alternative to visualize co-occurrences is an interactive, tree-based design [WV08] that allows co-occurrence frequencies in arbitrary window sizes. However, for our scenarios such a design is not viable because word embeddings only encode pairwise co-occurrences within a fixed size window.

Typical word embeddings have vocabularies of well over 1m words. A critical design decision is thus to select co-occurrences that help users with their tasks. Our design is based on the experience that it is often a set of words whose reconstructed co-occurrences are analyzed. Therefore, users can choose a set of words that they want to investigate. Based on these, we select informative reconstructed co-occurrences that provide insight into the similarities and differences between the selected words. We have experimented with absolute counts to select reconstructed co-occurrences, however, this results in the most common words to show up, even after filtering stop words.

An alternative strategy we found to work well is selecting the reconstructed co-occurrences with the highest variance between the words. For this, we create a list of reconstructed co-occurrence strengths for each selected word with each other word. We then calculate the variance of these reconstructed co-occurrences between the selected words for each embedding. The co-occurring words with the highest variance are selected and displayed as columns in the matrix, sorted by variance. Even though the process involves many vector products, the computation only takes a few seconds on a modern machine (see §5 for implementation details). While this strategy tends to yield informative reconstructed co-occurrences, users can freely add additional columns to the matrix based on their analysis goals.

Figure 4 shows the strongest reconstructed co-occurrences for the vectors *aunt*, *uncle*, *sister*, *brother*, *woman*, and *man*. Investigating the matrix, we get interesting additional insight that corroborates our hypothesis from §4.1. Again, we can see a lot of words that denote family relations. Perhaps not surprisingly, they co-occur more frequently with other words that denote such relations and that we have queried for. They appear less with the more general *man* and *woman*. In addition, we can see that female relatives co-occur more frequently with *aunt* and *sister*, while male ones occur more frequently with *uncle* and *brother*. On the other hand, *man* and *woman* does not show such a pattern. These gives us an idea about some of the patterns that lead to the findings from §4.1. In addition, we can see that, although not as pronounced, male and female nobility titles (including *king*, *prince*, and *duke*) show a similar reconstructed occurrence patterns than family relations. We hypothesize that this is due to the fact that wikipedia lists in par-

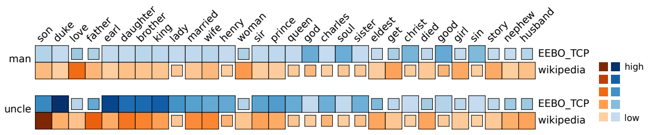

Figure 5: *Comparing the highest reconstructed co-occurrence differences between* man *and* uncle *across two different embeddings. Color encodes reconstructed co-occurrence strength. The fully filled matrix cells are those selected because of their high variance between the words (lines) in the corresponding embedding. All others are included for comparison. Nobility titles are more pronounced for* uncle*, especially in the EBBO-TCP embedding.*

ticular male family relations with nobility titles when discussing successions, thus revealing a bias of the underlying corpus.

The design also supports comparisons between multiple embeddings (see Figure 5). Each of the embeddings has a distinct color hue, while reconstructed co-occurrence strength is encoded as lightness. For each of the user-selected words, the reconstructed co-occurrences for each of the embeddings are shown in adjacent lines in the respective colors. Since a user-selected number of high-variance co-occurrences is extracted individually for each of the embeddings, some of those words can overlap. As a potential design alternative, we considered an interlaced version of the matrix, similar to [ABHR*] that encodes multiple values for different embeddings per cell. Due to the difficulty of encoding values for more than two embeddings, we decided in favor of the multi-row approach. Rows with reconstructed co-occurrences for different embeddings are grouped by user selected words, allowing for easy comparison of values between embeddings.

As the variance-based selection measure may yield different words for each of the embeddings, we show their union set. Words whose cells are not fully filled for a particular embedding are not among the selection. While using area is generally a more effective visual variable for continuous values, such as co-occurrence strengths, we found that using it for that purpose leads to matrix rows that seem to be discontinuous in their shape, rendering them harder to read and compare. We thus use it to encode a binary variable, while we use color for reconstructed co-occurrence strength.

Figure 5 shows an example where we have queried for two words, *man* and *uncle*. Both embeddings yield quite different patterns, with 10 overlapping words out of 20 extracted for each embedding. Based on our experience, 20 co-occurrences are a good first choice to get an overview. Interestingly, the nobility pattern is even stronger for this second embedding, with *uncle* having high co-occurrence strength with, for example, *duke* and *king*. While this is not surprising given the fact that EEBO-TCP contains historic texts that are more then 300 years old, it illustrates influences of the base copora on the embeddings.

4.3. Mapping to Concept Axes

To support **tasks 7 and 8**, analyzing and comparing word vector relations based on concept axes, we allow users to define their own axes to lay out word sets. Mapping a set of words to concept axes

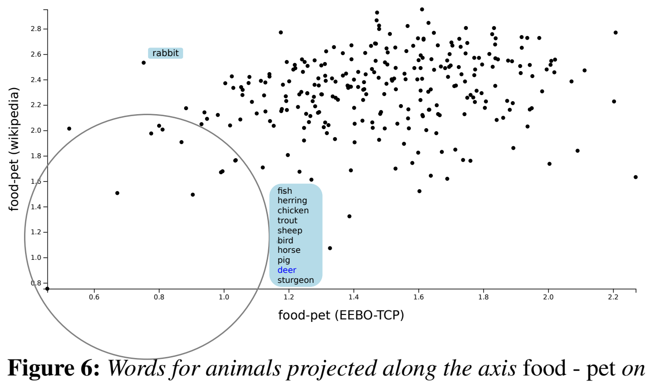

Figure 6: *Words for animals projected along the axis* food - pet *on two different embeddings. Exploring this space provides insights into the influences of different corpora on the resulting embedding. The lens facilitates exploration by summarizing words underneath it.*

conveys a sense of similarities within the embedding space. Unlike inspecting local neighborhoods, however, they provide a more global picture of similarity relations relative to two opposing concepts. Users may want to compare the same axes in two different embeddings, or different axes from the same embedding.

Our environment features a design based on the classical scatter plot to lay out word vectors along two axes. Scatter plots are used in some of the literature we reviewed to show linear projections, e.g., in [JD14]. While an alternative would be to use a combination of position and color encoding, similar to the one in §4.1, scatter plots are easier for gauging correlations between both axes, and identifying outliers. This is relevant to identify similarities and differences between the two axes. The idea of letting users define axes to lay out data for analysis has been explored before. Explainers [Gle13] and InterAxis [KCPE16] are both approaches that let users define such axes based on example data. While both approaches feature scatter plots to visualize such axes, the former introduces an alternative visualization that aligns data items and labels along a 1D axis. As this design does not facilitate comparison between two axes, we use 2D scatter plots.

Users can load a set of words and an embedding and select two words that represent opposing concepts to define an axis. For each axis, vectors of the selected word list are projected onto the difference vector between those concepts. Figure 6 shows the result of such an operation. In this example, both axes show the same concepts, while the embeddings they are based on differ. Both axes are labeled accordingly. In addition to visually analyzing correlations, there are additional features to look into local phenomena of the point clouds. Hovering an item with the mouse shows a tool tip with the respective word. This is shown in Figure 6 for the word *rabbit*.

For exploration, a magic lens [BSP*93] is available that summarizes the area of the scatter plot it hovers over [HJH*16], based on excentric labeling [FP99,BRL]. While an alternative method would be to statically label areas of the space, this lens-based interactive design gives users the flexibility to explore the space at various levels of granularity. It can be dragged across the space, showing the words underneath in a list next to it (see Figure 6). This list contains a maximum of around 10 words to keep it easily readable. Users can activate links between the words and their points, which are deactivated by default to reduce clutter (see Figure 10). If more than 10 words are hovered, clusters of similar ones are condensed into one. Clustered words in the list are marked in blue (see *deer* in Figure 6, which is clustered together with *cow* in the example), and clicking them reveals all words they summarize. For summarization, we use the word similarities based on the embedding, and replace a set of similar words with their centroid (see §5 for implementation details). While this has the downside of using the embedding we want to analyze for the abstractions, with the potential of distorting results, we have decided in favor of this heuristic because it does not require users to provide any additional source of data about the embeddings or its corpus. This renders the approach applicable much more broadly.

Figure 6 shows a scatter plot for the axis *food* − *pet* in our two embeddings. The selected word list contains 483 animals. Words towards the bottom of the plot are closer to the *food* concept in the wikipedia embedding, while those towards the left are closer to *food* in the EEBO-TCP embedding. Conversely, words towards the top and right of the plots are close to the *pet* concept in the respective embedding.

We have noticed differences in both embeddings with respect to food items, especially different types of meat. We are interested in analyzing this further and gain insights into how the underlying corpus contributes to this. Starting to investigate Figure 6, there seems to be a weak linear relationship between both dimensions with a large number of outliers. Using the lens to investigate this further, we can see that the bottom left of the space contains animals that are typically used as food, including *fish* and *chicken*, but also less clear cases such as *horse* and *bird*. Looking into those, we can see that both are in the top right corner of the hovered space. One apparent outlier on the top catches our attention. Hovering it reveals that it is the word *rabbit*. We hypothesize that while being a source of food in the past, rabbits nowadays are often kept as pets. Continuing with this analysis, we can learn more about the biases that corpus selection has on resulting embeddings.

5. Implementation

The implementation is based on common web technologies. All embedding processing is done within a Python-based (Python 3.6 with numpy) back end, providing interactive response times. Although our implementation is model agnostic, our examples were created with GloVe or word2vec. We use the flask web framework to interface with the front end in the browser, which is implemented in javascript (using d3.js and jQuery UI). Our implementation is available as open source software, including a docker image for simple deployment. Links are available on the project page ‖. It can be deployed locally or publicly, e.g., using virtual server infrastructure.

One challenging implementation aspect was to ensure interactive response times for the lens (§4.3). To retrieve and summarize words efficiently, we use a quadtree to speed up both processing for a given 2D projection. To ensure the list is no longer than ten words, we replace sets of words with their centroids in the embeddings, stored in intermediary nodes of the tree. These abstractions are attached to the quadtree in the server and sent to the client once for each plot. This process yields interactive rendering rates for smooth exploration.

‖ https://graphics.cs.wisc.edu/Vis/EmbVis

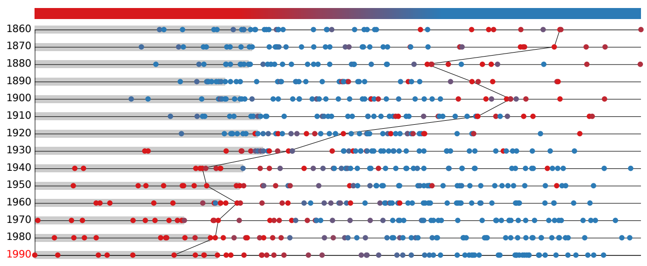

Figure 7: *Nearest neighbors of the word* broadcast *in multiple word embeddings trained on corpora from different decades. The 1990s embedding is selected and the color on each axis encodes distance on the 1990s axis. Prior to 1930, words close to broadcast in the 1990s (red marks) are farther from it. The word* stations *is selected and linked across the axes. The color ramp is chosen for clarity in print.*

6. Use Cases

6.1. Changes on Word Meaning

Word embeddings are used to study changes in word meaning over time, for example by Hamilton et al. [HLJ16]. For their project, they train multiple embeddings on the English Google Ngram corpus that contains ngrams of books in the Google book database from the 16th to the 21st century. We use their embeddings trained for each decade from 1800 to 1990 (with skip-gram word2vec [MSC*13]). Here, we show how to use our methods to quickly get insight into how a word's meaning changed and whether we can find evidence for this change in the embeddings.

One particularly interesting case to look at is the word *broadcast* [HLJ16]. Its use until the early 20th century denoted sowing a field by scattering seeds. With the introduction of radio technology, its meaning changed to the distribution of information through electromagnetic waves [JL79]. We investigate this change using our neighborhood comparison method with embeddings from 1860 to 1990 (earlier ones do not contain the word *broadcast*). This results in a plot that reveals a fair amount of new words that enter the neighborhood of *broadcast* until the early 20th century, based on the color of some of the circles. Since many of the changes are technical terms, we hypothesize that these are due to progress in sowing technology at that time.

Inspecting the neighborhoods more closely, we identify the decade of the 1920s as a potentially major turning point. We can see that almost all of the neighbors from the 1920 embedding get replaced in the embedding from the 1930s. This coincides with the history of radio transmission, which started in the 1920s (Hamilton et al. [HLJ16] also identify this turning point).

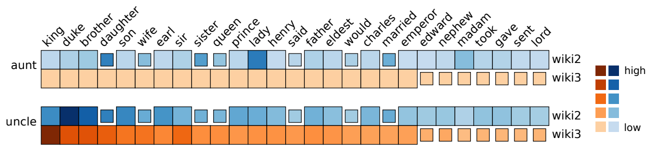

Figure 8: *Comparing reconstructed co-occurrences between embedding 2 and 3 trained on the wikipedia data set. Color encodes co-occurrence strength. The fully filled matrix cells are those selected because of their high variance between the words (lines) in the corresponding embedding. All others are included for comparison. Generally,* aunt *has very low and uniform co-occurrence values in embedding 3, compared to embedding 2.*

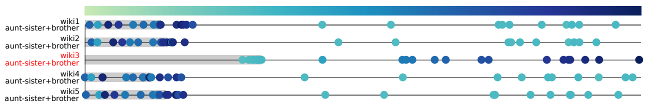

Figure 9: *Comparing the neighborhoods of* $vec(aunt) - vec(sister) + vec(brother)$ *across five embeddings trained with the same parameters and on the same corpus (wikipedia). One of the embeddings, embedding 3 (selected) shows a very different neighborhood than all others.*

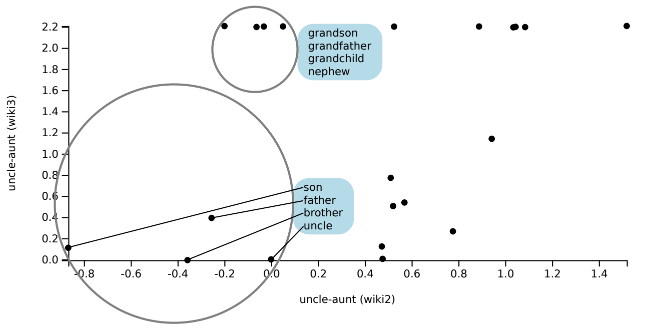

Figure 10: *Comparing family relations along the* uncle - aunt *axis in embedding 2 and 3. For the figure, two lenses are shown in the same plot. The lower one has links between words and their points activated. Surprisingly, there is no correlation between word positions in both embeddings.*

To get a better picture of the entire time line, we select the 1990s as the reference point for all axes (see Figure 7). Since *broadcast* has almost completely lost its original meaning today [JL79], this gives us decades in which it had a roughly similar meaning to today, and ones in which it had a different one. We can see a clear separation between the decades before 1920, and those after it. In addition, the 1930s seem to have had more changes compared to 1990 than later decades. Inspecting some of the words that later leave its neighborhood reveal that they are all related to *sowing*, while the one that moves significantly closer is the term *television*, which was invented in the 1930s. Another interesting case is the word *stations*, which is selected and linked by lines across the axes in Figure 7. It moves into the neighborhood of *broadcast* from a place far out.

6.2. Stability of Embeddings

One common concern about word embeddings is the non-determinism of embedding methods, resulting in unstable neighborhoods [HH16]. Our tools can be used to assess this issue. We explore the examples of the previous section by creating five embeddings using the same algorithm (GloVe) and window size (15). This strategy is similar to the one that Hellrich and Hahn [HH17] use for their study.

We start by comparing the neighborhoods of words in the different embeddings using *uncle* from §4 as the example. The result shows that four of the embeddings have nearly identical neighborhoods, but that there is one outlier, embedding 3. Interestingly, it has very little overlap with the neighborhoods from the other embeddings. Inspecting the exceptions reveals that those are *brother*, *sister*, and *daughter*. Looking at the five neighborhoods of *aunt* reveals that the situation is similar for it, with no overlap in its neighborhood between embedding 3 and the others. Being trained on the same corpus and with the same parameter settings, these differences seem surprising.

To investigate these differences further, we look at the encoded co-occurrences between *aunt* and *uncle*, which shows only minor differences between embeddings 1, 2, 4 and 5. When comparing each of them to embedding 3, however, we see major differences. This is illustrated by the comparison of embedding 2 with 3 based on *uncle* and *aunt* as shown in Figure 8. In embedding 2, as we would expect, *aunt* appears more in the context of female honorifics (*lady*, *madam*), and female family members (*sister*, *daughter*). However, in embedding 3, *aunt*'s reconstructed co-occurrences look very uniform. In comparison, while the encoded co-occurrences of *uncle* also differ to some extent between both embeddings, coinciding with the differing neighborhoods discussed previously, the differences are much less pronounced and within a conceivable range.

We wonder how these differences influence arithmetic structures based on these vectors. For this, we use the analogy from §4: $vec(aunt) - vec(sister) + vec(brother)$. Comparing neighborhoods, embedding 3 again stands out. While all other embeddings yield *uncle* as the nearest neighbor, and have very little variance in terms of their neighborhoods, the nearest neighborhood in embedding 3 is *exciseable* follow by *zealously*, two apparently unrelated and unexpected words. Figure 9 shows the neighborhood plot, comparing embedding 3 (selected) to all others.

For a more global picture of differences, we compile a list of 21 family relations, and lay them out according to the *uncle* - *aunt* axis in embedding 2 and 3 (see Figure 10). We would expect male and female terms to spread out along this axis, with male terms closer to *uncle*, and female terms closer to *aunt*, with a high level of correlation between both axes. Surprisingly, there is hardly any correlation visible. When exploring the plot, we find that for embedding 2, our expectations are largely met. In addition to the male terms in the lower left corner close to *uncle* in both embeddings, female terms are found further to the right for embedding 2. For embedding 3, however, we find a cluster of outliers on top of the plot, where we would expect female terms.

These findings illustrate that the stochastic nature of training embeddings can result in unexpected and unwanted local results. While we should be careful to generalize these local issues and draw conclusions about the overall quality of the embedding, uncovering and analyzing problematic local properties can help to better choose between and debug multiple embeddings for a specific application scenario.

7. Discussion

In this paper, we derived a comprehensive task space for the visual analysis of word vector embeddings. In addition, we provided and discussed multiple interactive visual designs that cover large parts of this space. While we rely heavily on domain literature for this process, we have also discussed our designs with experts from the DH and NLP communities. This happened at various design and implementation stages to validate our design rationales, and get feedback on the effectiveness and usability of the designs. While these sessions were conducted in an informal setting, we still report some key insights as a basis for discussion.

We held sessions with three experts, two of which are active literary researchers with an interest in but no prior experience with word embeddings. One of the literary scholars had prior experience with programming and computer science. The third expert is an active NLP researcher working on embedding algorithms. While earlier demo and feedback sessions were based on screen shots and static images of the designs, later ones included fully functional prototypes that could be used and tested by the experts. During each of the different stages, we used the feedback and inspiration we received from the experts to refine the designs and implementations. This includes improving usability of the prototypes as well as increasing their effectiveness for various analysis scenarios. Each of the sessions included an introduction to the visualizations, and demos of the use cases we derived from the literature. We then asked the experts about the usability of the implementations, and if and how they would use the designs in context of their work. These questions were the starting point of an open ended discussion about the designs, and the problems from their domains that the experts were interested in.

All of the experts provided positive feedback on each of the visualizations during the sessions. Their interest in them, however, differed depending on their background. Both groups were interested in neighborhood comparison, the literary experts had an additional interest in concept axes. The NLP researcher's focus, instead, found the reconstructed co-occurrences important, and mentioned that they are useful for debugging and comparing multiple embeddings. While this shows that the tasks covered by our designs are overall relevant to domain experts, this result is also little surprising given the literature of different communities the respective tasks are motivated by (see supplemental material for more information). Inspecting and comparing neighborhoods is a popular task in both domains, while the motivation for analyzing co-occurrences is largely based on NLP literature. On the other hand, mapping word vectors to concept axes is strongly motivated from the DH community.

The experts found each of the designs easy to understand and interpret after a brief introduction. While this was also true for the neighborhood comparison visualization, one of them remarked that, although he generally liked it, understanding the visual encoding might be too complicated for non-technically inclined persons. As we target users that use word embeddings in their work, we can presuppose some technical expertise. However, this underlines the importance of straightforward visual representations to increase effectiveness.

While using the implementations and engaging in discussions, the experts were able to explore questions that they were interested in. These included analyzing the historical metaphorical use of animal terms for such concepts as *female*, *male*, or *devil* or the exploration of cultural overtones in food terms. Most of the examples described in the previous section, where synthesized or influenced by these examples and discussions, including both use cases.

While we cover the majority of the task space with the designs, we lack tools that provide a more global view that will help identify the appropriate words, synonym groups, and analogies to explore in detail with the methods presented here. Potentially interesting structures, for example, include densely clustered words within the embedding, or salient vector offsets that are likely to convey meaning. In addition, the problem of visually comparing those structures between embeddings is another open and important challenge. For both challenges, labeled resources that are used to evaluate embeddings can play an important role in guiding users towards interesting areas in the embedding. This is an open research problem and an important direction for future work.

Another future direction is the extension of our tasks and techniques beyond the scope of word embeddings. While targeting embeddings of other textual entities, such as phrases or sentences is an interesting goal, there are many types of data beyond text for which embedding techniques are used. Examples include images, videos, or graphs. Embeddings of those types of data are ubiquitous in machine learning and data analysis scenarios, and effective visual methods to analyze and compare them will help improve relevant domain tasks. While we expect most of our practical embedding tasks and some of our methods to be transferable to these domains, they provide new domain and data-specific tasks and visualization challenges.

Acknowledgments

We acknowledge the reviewers for their thoughtful comments and suggestions, as well as our three domain experts for their invaluable feedback and comments on our work. This work was in part supported by NSF award 1162037 and DARPA FA8750-17-2-0107.

References

[ABHR*] ALPER B., BACH B., HENRY RICHE N., ISENBERG T., FEKETE J.-D.: Weighted graph comparison techniques for brain connectivity analysis. 8

[AG16] ALEXANDER E., GLEICHER M.: Task-driven comparison of topic models. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (January 2016), 320–329. 2, 6

[AKV*14] ALEXANDER E., KOHLMANN J., VALENZA R., WITMORE M., GLEICHER M.: Serendip: Topic model-driven visual exploration of text corpora. In *IEEE Conference on Visual Analytics Science and Technology* (2014), IEEE, pp. 173–182. 2

[ALL*16] Arora S., Li Y., Liang Y., Ma T., Risteski A.: A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics 4* (2016), 385–399. 2

[ALM17] Arora S., Liang Y., Ma T.: A simple but tough-to-beat baseline for sentence embeddings. In *Proc. of ICLR* (2017). 2

[BCZ*16] Bolukbasi T., Chang K.-W., Zou J. Y., Saligrama V., Kalai A. T.: Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Neural Information Processing* (2016), pp. 4349–4357. 3, 4

[BDK14] Baroni M., Dinu G., Kruszewski G.: Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proc. of ACL* (2014), pp. 238–247. 3, 4

[BGH*17] Bowman S. R., Goldberg Y., Hill F., Lazaridou A., Levy O., Reichart R., Søgaard A. (Eds.):. *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP, RepEval@EMNLP 2017, Copenhagen, Denmark, September 8, 2017* (2017), Association for Computational Linguistics. 1

[BMS17] Berger M., McDonough K., Seversky L. M.: cite2vec: Citation-driven document exploration via word embeddings. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan 2017), 691–700. 3

[BRL] Bertini E., Rigamonti M., Lalanne D.: Extended excentric labeling. In *EuroVIS'09*. 3, 8

[BSP*93] Bier E. A., Stone M. C., Pier K., Buxton W., DeRose T. D.: Toolglass and magic lenses: The see-through interface. In *Proc. of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1993), SIGGRAPH '93, ACM, pp. 73–80. 8

[CCP09] Collins C., Carpendale S., Penn G.: Docuburst: Visualizing document content using language structure. *Computer Graphics Forum 28*, 3 (2009), 1039–1046. 3

[CLRP13] Choo J., Lee C., Reddy C. K., Park H.: Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec 2013), 1992–2001. 2

[CVW09] Collins C., Viegas F. B., Wattenberg M.: Parallel tag clouds to explore and analyze faceted text corpora. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on* (2009), IEEE, pp. 91–98. 6

[FCB16] Fast E., Chen B., Bernstein M. S.: Empath: Understanding topic signals in large-scale text. In *Proc. of CHI* (2016), pp. 4647–4657. 1, 2

[FDJ*14] Faruqui M., Dodge J., Jauhar S. K., Dyer C., Hovy E., Smith N. A.: Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166* (2014). 3, 4

[FP99] Fekete J.-D., Plaisant C.: Excentric labeling: Dynamic neighborhood labeling for data visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 1999), CHI '99, ACM, pp. 512–519. 3, 8

[FRMW17] Fulda N., Ricks D., Murdoch B., Wingate D.: What can you do with a rock? affordance extraction via word embeddings. *arXiv preprint arXiv:1703.03429* (2017). 3

[GD16] Gladkova A., Drozd A.: Intrinsic evaluations of word embeddings: What can we do better? In *Proc. of the 1st Workshop on Evaluating Vector-Space Representations for NLP* (2016), pp. 36–42. 5

[Gle13] Gleicher M.: Explainers: Expert explorations with crafted projections. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec 2013), 2042–2051. 2, 8

[Gle16] Gleicher M.: A framework for considering comprehensibility in modeling. *Big data 4*, 2 (2016), 75–88. 2

[HH16] Hellrich J., Hahn U.: Bad company-neighborhoods in neural embedding spaces considered harmful. In *COLING* (2016), pp. 2785–2796. 10

[HH17] Hellrich J., Hahn U.: Don't get fooled by word embeddings: better watch their neighborhood. In *Conference of the Alliance of Digital Humanities Organizations (Abstracts)* (2017). 10

[HJH*16] Heimerl F., John M., Han Q., Koch S., Ertl T.: Docucompass: Effective exploration of document landscapes. In *IEEE Conference on Visual Analytics Science and Technology* (Oct 2016), pp. 11–20. 3, 8

[HKBE12] Heimerl F., Koch S., Bosch H., Ertl T.: Visual classifier training for text document retrieval. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (Dec 2012), 2839–2848. 2

[HLJ16] Hamilton W. L., Leskovec J., Jurafsky D.: Diachronic word embeddings reveal statistical laws of semantic change. In *Proc. of ACL* (2016), pp. 1489–1501. 1, 2, 3, 4, 9

[JD14] Jatowt A., Duh K.: A framework for analyzing semantic change of words across time. In *Proc. of JCDL* (2014), pp. 229–238. 8

[JL79] Jeffers R. J., Lehiste I.: *Principles and Methods for Historical Linguistics*, 1 ed. The MIT Press, 1979. 9, 10

[KCPE16] Kim H., Choo J., Park H., Endert A.: Interaxis: Steering scatterplot axes via observation-level interaction. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (Jan 2016), 131–140. 2, 8

[Kes17] Kessler J. S.: Scattertext: a browser-based tool for visualizing how corpora differ. *CoRR abs/1703.00565* (2017). 3

[KSKW15] Kusner M., Sun Y., Kolkin N., Weinberger K.: From word embeddings to document distances. In *International Conference on Machine Learning* (2015), pp. 957–966. 1

[LBT*17] Liu S., Bremer P. T., Thiagarajan J. J., Srikumar V., Wang B., Livnat Y., Pascucci V.: Visual exploration of semantic relationships in neural word embeddings. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (2017), 1–1. 1, 2, 5

[LG14] Levy O., Goldberg Y.: Linguistic regularities in sparse and explicit word representations. In *Proc. of CoNLL* (2014), pp. 171–180. 1, 3

[LHK*16] Levy O., Hill F., Korhonen A., Cho K., Reichart R., Goldberg Y., Bordes A. (Eds.):. *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP, RepEval@ACL 2016, Berlin, Germany, August 2016* (2016), Association for Computational Linguistics. 1

[LJW*15] Liu Q., Jiang H., Wei S., Ling Z.-H., Hu Y.: Learning semantic word embeddings based on ordinal knowledge constraints. In *Association for Computational Linguistics* (2015), pp. 1501–1511. 3, 4

[LSC*17] Liu M., Shi J., Cao K., Zhu J., Liu S.: Analyzing the training processes of deep generative models. *IEEE Transactions on Visualization and Computer Graphics* (2017). 2

[LSL*17] Liu M., Shi J., Li Z., Li C., Zhu J., Liu S.: Towards better analysis of deep convolutional neural networks. *IEEE transactions on visualization and computer graphics 23*, 1 (2017), 91–100. 2

[MCCD13] Mikolov T., Chen K., Corrado G., Dean J.: Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013). 1, 2, 4, 5

[MH08] Maaten L. v. d., Hinton G.: Visualizing data using t-sne. *Journal of Machine Learning Research 9*, Nov (2008), 2579–2605. 2

[MLMP17] M/"uhlbacher T., Linhardt L., Möller T., Piringer H.: Treepod: Sensitivity-aware selection of pareto-optimal decision trees. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (2017), 1–1. 2

[MSC*13] Mikolov T., Sutskever I., Chen K., Corrado G., Dean J.: Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS* (2013), pp. 3111–3119. 2, 9

[Mun14] Munzner T.: *Visualization analysis and design*. CRC press, 2014. 3, 4

[NAM16] NAYAK N., ANGELI G., MANNING C. D.: Evaluating word embeddings using a representative suite of practical tasks. In *Proc. of the ACL Workshop on Evaluating Vector-Space Representations for NLP* (2016), pp. 31–35. 3

[PKL*17] PARK D., KIM S., LEE J., CHOO J., DIAKOPOULOS N., ELMQVIST N.: Conceptvector: Text visual analytics via interactive lexicon building using word embedding. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (2017), 1–1. 3

[PSM14] PENNINGTON J., SOCHER R., MANNING C. D.: Glove: Global vectors for word representation. In *Proc. of EMNLP* (2014), pp. 1532–1543. 1, 2, 3, 4

[RA16] RONG X., ADAR E.: Visual tools for debugging neural language models. In *Proc. of ICML Workshop on Visualization for Deep Learning* (2016). 2

[RBS17] ROSSIELLO G., BASILE P., SEMERARO G.: Centroid-based text summarization through compositionality of word embeddings. In *Proc. of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres* (04 2017), pp. 12–21. 3, 4

[RG65] RUBENSTEIN H., GOODENOUGH J. B.: Contextual correlates of synonymy. *Communications of the ACM 8*, 10 (Oct. 1965), 627–633. 2

[SBM*13] SOCHER R., BAUER J., MANNING C. D., ET AL.: Parsing with compositional vector grammars. In *Proc. of the 51st Annual Meeting of the Association for Computational Linguistics* (2013), vol. 1, pp. 455–465. 1

[SGL08] STASKO J., GÖRG C., LIU Z.: Jigsaw: supporting investigative analysis through interactive visualization. *Information visualisation 7*, 2 (2008), 118–132. 2

[SGPR17] STROBELT H., GEHRMANN S., PFISTER H., RUSH A. M.: Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics* (2017). 2

[Shn96] SHNEIDERMAN B.: The eyes have it: A task by data type taxonomy for information visualizations. In *Proc. of the IEEE Symposium on Visual Languages* (1996), IEEE, pp. 336–343. 5

[SLMJ15] SCHNABEL T., LABUTOV I., MIMNO D., JOACHIMS T.: Evaluation methods for unsupervised word embeddings. In *Proc. of EMNLP* (2015), pp. 298–307. 3

[SMM12] SEDLMAIR M., MEYER M., MUNZNER T.: Design study methodology: Reflections from the trenches and the stacks. *IEEE transactions on visualization and computer graphics 18*, 12 (2012), 2431–2440. 3

[SN16] SUZUKI J., NAGATA M.: Right-truncatable neural word embeddings. In *HLT-NAACL* (2016), pp. 1145–1151. 3

[SNHS13] SCHULZ H.-J., NOCKE T., HEITZLER M., SCHUMANN H.: A design space of visualization tasks. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec. 2013), 2366–2375. 4, 5

[STN*16] SMILKOV D., THORAT N., NICHOLSON C., REIF E., VIÉGAS F. B., WATTENBERG M.: Embedding projector: Interactive visualization and interpretation of embeddings. *arXiv preprint arXiv:1611.05469* (2016). 2

[TLKT] TALBOT J., LEE B., KAPOOR A., TAN D. S.: Ensemblematrix: Interactive visualization to support machine learning with multiple classifiers. 2

[vdEvW11] VAN DEN ELZEN S., VAN WIJK J. J.: Baobabview: Interactive construction and analysis of decision trees. In *IEEE Conference on Visual Analytics Science and Technology* (Oct 2011), pp. 151–160. 2

[vHWV09] VAN HAM F., WATTENBERG M., VIEGAS F. B.: Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov 2009), 1169–1176. 3

[WLL*16] WANG X., LIU S., LIU J., CHEN J., ZHU J., GUO B.: Topicpanorama: A full picture of relevant topics. *IEEE Transactions on Visualization and Computer Graphics 22*, 12 (Dec 2016), 2508–2521. 2

[WSW*17] WONGSUPHASAWAT K., SMILKOV D., WEXLER J., WILSON J., MANÃL' D., FRITZ D., KRISHNAN D., VIÃL'GAS F. B., WATTENBERG M.: Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (2017), 1–1. 2

[WV08] WATTENBERG M., VIÃL'GAS F. B.: The word tree, an interactive visual concordance. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1221–1228. 3, 7

[YS16] YIN W., SCHÜTZE H.: Learning word meta-embeddings. In *Proc. of ACL* (2016), pp. 1351–1360. 3

[YWL*16] YIN R., WANG Q., LI P., LI R., WANG B.: Multi-granularity chinese word embedding. In *EMNLP* (2016), pp. 981–986. 3, 4

[YWLZ17] YU L.-C., WANG J., LAI K. R., ZHANG X.: Refining word embeddings for sentiment analysis. In *EMNLP* (2017), pp. 545–550. 1

[ZSCM13] ZOU W. Y., SOCHER R., CER D., MANNING C. D.: Bilingual word embeddings for phrase-based machine translation. In *EMNLP* (2013), pp. 1393–1398. 2