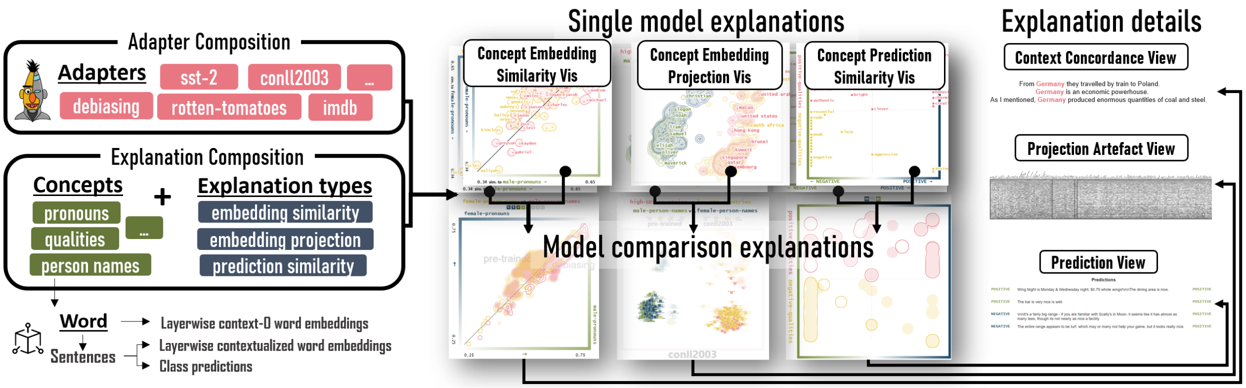
Visual Comparison of Language Model Adaptation

Rita Sevastjanova, Eren Cakmak, Shauli Ravfogel, Ryan Cotterell, and Mennatallah El-Assady



From Germany they travelled by train to Poland.
Germany is an economic powerhouse.
As I mentioned, Germany produced enormous quantities of coal and steel.

Fig. 1: We present a workspace that enables the evaluation and comparison of adapters – lightweight alternatives for language model fine-tuning. After data pre-processing (e.g., embedding extraction), users can select pre-trained adapters, create explanations, and explore model differences through three types of visualizations: *Concept Embedding Similarity*, *Concept Embedding Projection*, and *Concept Prediction Similarity*. The explanations are provided for single models as well as model comparisons. For each explanation, we provide further explanation details, such as the word contexts as well as embedding vectors themselves.

Abstract—Neural language models are widely used; however, their model parameters often need to be adapted to the specific domains and tasks of an application, which is time- and resource-consuming. Thus, adapters have recently been introduced as a lightweight alternative for model adaptation. They consist of a small set of task-specific parameters with a reduced training time and simple parameter composition. The simplicity of adapter training and composition comes along with new challenges, such as maintaining an overview of adapter properties and effectively comparing their produced embedding spaces. To help developers overcome these challenges, we provide a twofold contribution. First, in close collaboration with NLP researchers, we conducted a requirement analysis for an approach supporting adapter evaluation and detected, among others, the need for both intrinsic (i.e., embedding similarity-based) and extrinsic (i.e., prediction-based) explanation methods. Second, motivated by the gathered requirements, we designed a flexible visual analytics workspace that enables the comparison of adapter properties. In this paper, we discuss several design iterations and alternatives for interactive, comparative visual explanation methods. Our comparative visualizations show the differences in the adapted embedding vectors and prediction outcomes for diverse human-interpretable concepts (e.g., *person names*, *human qualities*). We evaluate our workspace through case studies and show that, for instance, an adapter trained on the language debiasing task according to context-0 (*decontextualized*) embeddings introduces a new type of bias where words (even gender-independent words such as countries) become more similar to female- than male pronouns. We demonstrate that these are artifacts of context-0 embeddings, and the adapter effectively eliminates the gender information from the contextualized word representations.

Index Terms—Language Model Adaptation, Adapter, Word Embeddings, Sequence Classification, Visual Analytics

1 Introduction

Language models (LMs) such as the masked language model BERT [11] are widely used for diverse natural language processing (NLP) and understanding tasks. Such models are capable of learning manifold language properties in an unsupervised manner [59]. However, the model parameters typically need to be updated before using them on downstream tasks, such as sentiment classification. Task specific fine-tuning [27,55] along with domain specific fine-tuning [21,22] are the most common methods for parameter adaptation. Although fine-tuning methods commonly achieve state-of-the-art results on many NLP tasks [55], they come along with limitations such as a high training time and storage [32]. To overcome the shortcomings of the model fine-tuning, Houlsby et al. [26] have recently introduced adapter modules – a lightweight alternative for LM fine-tuning. Instead of adapting the complete model, adapters learn a small set of task-specific parameters, requiring less training time and storage space. For a more efficient adapter training and composition, Pfeiffer et al. [49] have proposed a modular adapter framework called AdapterHub. It comes along with adapter-transformers – an extension of HuggingFace's transformers library[1], integrating adapters into state-of-the-art LMs. In addition to the simple parameter adaptation, the AdapterHub framework allows sharing adapters with the community, supporting open science practices.

The AdapterHub repository currently contains almost 400 adapters for 72 text analysis tasks and 50 languages. To select the best adapter for a given analysis task, one needs to be able to compare the adapters and their learned language properties. The related work has shown that such model comparison tasks are the focus of both model- and data-driven users working with LMs [5]. To understand more about the typical analysis setting, data, and performed tasks when evaluating fine-tuned model properties, we conducted literature review and semi-

- *Rita Sevastjanova and Eren Cakmak are with University of Konstanz. E-mail: firstname.lastname@uni-konstanz.de.*
- *Shauli Ravfogel is with Bar-Ilan University. E-mail: shauli.ravfogel@gmail.com.*
- *Ryan Cotterell is with ETH. E-mail: ryan.cotterell@inf.ethz.ch.*
- *Mennatallah El-Assady is with ETH, AI Center. E-mail: melassady@ethz.ch.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

[1] https://github.com/Adapter-Hub/adapter-transformers

structured interviews with two NLP researchers. The requirement analysis revealed that researchers are interested in analyzing models with respect to different human-interpretable concepts. In particular, they investigate how specific concept representations change during fine-tuning. The analysis is typically performed on two types of data: (1) word embedding representations and (2) classifier prediction outcomes. Using word embeddings, they analyze evolving concept intersections as well as newly produced artifacts like strange word associations (e.g., biases). Prediction outcomes are used to analyze task-adapted model behavior changes, e.g., whether specific word associations lead to unexpected prediction outcomes.

The adapters trained on one particular task typically have different architectures [26, 50] and training corpora. These different learning settings usually lead to different model performances; it is difficult, though, to keep track of such performance variations. The continuous development of new adapters thus dictates the need for a solution that assists the analysis and comparison of adapter properties.

To support the NLP community in an effective adapter evaluation and comparison, we contribute a novel visual analytics workspace. The workspace integrates adapters from the AdapterHub repository and enables their analysis through three types of visual explanation methods: *Concept Embedding Similarity*, *Concept Embedding Projection*, and *Concept Prediction Similarity* (see Fig. 1). We support model comparison according to their produced word embeddings and classification predictions, i.e., both intrinsic and extrinsic evaluation methods. The explanations are performed on diverse human-interpretable concepts related to bias mitigation and sentiment analysis tasks (e.g., *gender-related stereotypes*, *human qualities*). The users can upload further concepts to the workspace to cover further analysis directions. The modular composition of visual explanations supports such analysis extensions.

The comparison of adapter properties requires sufficient comparative visualization designs. As described by Gleicher [19], the design of comparative visualizations is not trivial since they typically combine the issues of representing individual objects as well as their relationships. In order to design an appropriate solution, we rely on the comparative visualization guidelines [19] and consider four task- and data-related aspects: (1) comparative elements, (2) challenges related to representing relationships between the comparative elements, (3) strategies to overcome the challenges, and (4) a sufficient design solution. The design process constituted of several iterations in close collaboration with NLP researchers. In Sect. 5 we present some of the considered design alternatives; others are provided as supplementary material to this paper.

We show the applicability of the workspace through case studies created collaboratively with NLP researchers. In particular, we compare the properties of six adapters related to debiasing, sentiment classification, and named entity recognition tasks. We present new insights into model properties related to human-interpretable concepts and show that, for instance, context-0 (*decontextualized*) embeddings of the adapter trained on the language debiasing task contain a bias where words become more similar to *female-* than *male pronouns*; however, the gender information is eliminated from the contextualized word representations.

To summarize, the contribution of this paper is threefold. (1) We present requirements for a visual analytics system supporting fine-tuned LM comparison. (2) We introduce a workspace for model comparison and present design considerations for three types of comparative, visual explanation methods. (3) We present new insights into multiple adapter properties through expert case studies.

2 Background and Related Work

In the following, we describe background information related to LM fine-tuning and related work to explanation methods.

2.1 Language Model Fine-Tuning

In this paper, we analyze transformers, which are multi-layer models that use attention mechanisms [69]. In these models, each token of the input sequence is mapped to a high-dimensional vector (i.e., context-dependent embedding that encodes specific context properties). These embeddings are updated in each transformer's layer; thus, one can extract and analyze contextualized word embeddings layerwise (e.g., 12 layers for the BERT-base model). It has been shown that these embeddings encode different language properties found in the training data [59]. LMs, including transformers, are commonly fine-tuned to capture language characteristics for specific domains or tasks. Domain-adaptive fine-tuning is an unsupervised fine-tuning approach based on a masked language modeling task on text from a specific domain [22]. Intermediate-task training is a model's fine-tuning on labeled data prior to task-specific fine-tuning [52]. Task-specific fine-tuning deals with adapting an LM to a particular output label distribution [27]. The fine-tuning of LMs is effective yet time- and resource-consuming. Kirkpatrick et al. [32] also showed that fine-tuning can lead to catastrophic forgetting of language characteristics acquired during the model's pre-training. To overcome these limitations, Houlsby et al. [26] introduced adapters. They are a lightweight alternative for model fine-tuning, only optimizing a small set of task-specific parameters learned and stored during the adaptation phase, thus, reducing both training time and storage space. The AdapterHub framework [49] has brought the advantage of a simple and efficient adapter composition and reuse – one can upload their trained adapters to the AdapterHub or HuggingFace[2] repositories, and they are available in the framework for interested parties, supporting the open science practice. Adapters can be trained on masked language modeling as well as specific downstream tasks (e.g., sentiment classification). The trained adapters can be 'attached' to the pre-trained model, leading to adapted model parameters. The model with an attached task adapter can be used for the target task (e.g., sentiment classification). Adapters have been applied for tasks such as natural language generation [38], machine translation [31, 53], domain adaptation [18, 51], injection of external knowledge [35], and language debiasing [34].

2.2 Visual Embedding Explanation and Comparison

With respect to explainability, most relevant work has focused on visualizations that show **how** transformers work and **what** they learn. For example, visual analytics systems like NLIZE [40], Seq2Seq-Vis [66], BertViz [70], exBERT [25], SANVis [46], and Attention Flows [10] visualize the attention layer, i.e., to highlight tokens to which the model attends to in order to solve a task. Although widely used, attentions and their suitability for explanation purposes are being controversially discussed in related work (see, e.g., [28]). Other work has focused on visualizing word embeddings to show what LMs learn. The first such tools were designed for static embeddings, such as word2vec [44] and GloVe [47], and facilitated analogies [39] and tasks related to local word neighborhoods [23]. Later, Berger [3] explored correlations between embedding clusters in BERT [11]. Recent tools focus on LM comparison tasks by visualizing multiple models simultaneously. For instance, Strobelt et al. [67] present LMDiff – a tool that visually compares LM probability distributions and suggests interesting text instances for the analysis. Heimerl et al. [24] present embComb, which applies different metrics to measure differences in the local structure around embedding objects (e.g., tokens). Embedding Comparator by Boggust et al. [5] is a system for embedding comparison through small multiples. It calculates and visualizes similarity scores for the embedded objects based on their local neighborhoods (i.e., shared nearest neighbors). Different from these two approaches, we provide explanations of pre-defined human-interpretable concepts, enabling testing more specific hypotheses related to embedding intersections. Sivaraman et al. [65] present Emblaze, which uses an animated scatterplot and integrates visual augmentations to summarize changes in the analyzed embedding spaces. In contrast, we compare models by aligning the two spaces using juxtaposition, superposition, and explicit encoding techniques. Our recent work called LMFingerprints [62] applies scoring techniques to examine properties encoded in embedding vectors and supports model as well as model layer comparison. Embedding comparison tasks are relevant for all types of data that get represented by embedding vectors. For instance, Li et al. [36] present a visual analytics system for node embedding comparison (i.e., graph data), and Arendt et al. [1] introduce a visualization technique called Parallel Embeddings for concept-oriented model comparison on image data, to name a few.

[2] https://huggingface.co/

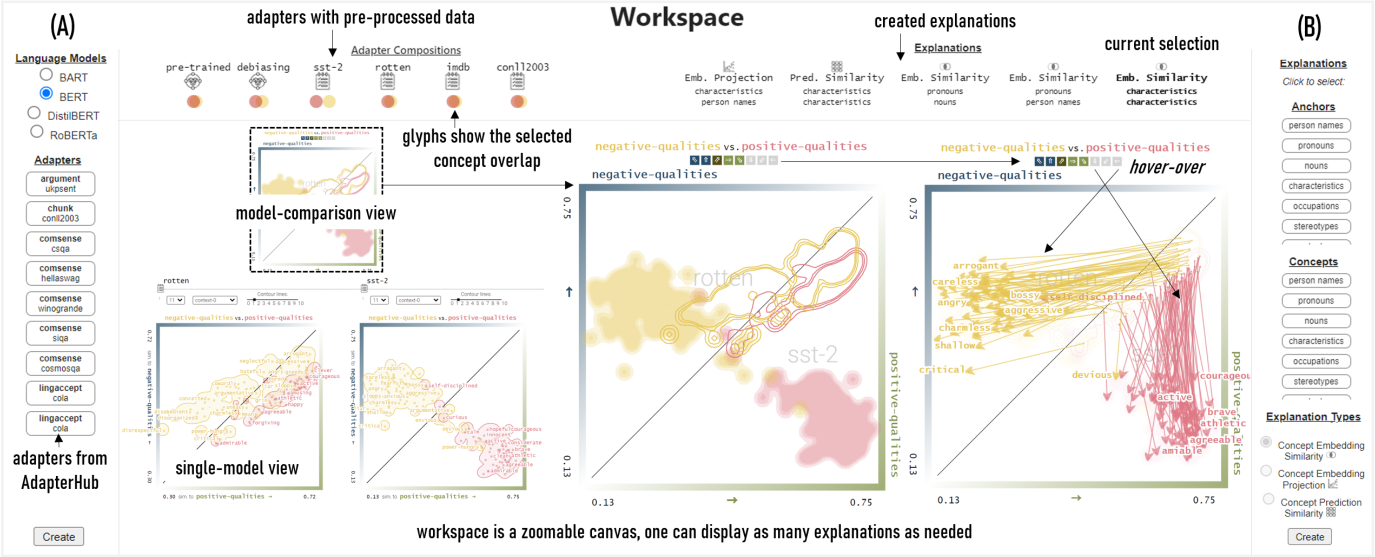


Fig. 2: The workspace contains three views: **Adapter Composition View** (A), which lists adapters from AdapterHub repository, **Explanation Composition View** (B) for modular explanation generation, and **Visual Comparison View** (Workspace) for model comparison. Here: contrary to the rotten-tomatoes model, the context-0 embeddings of the sst-2 sentiment classifier strongly encode the two polarities of *human qualities*.

3 Requirement Analysis

Before designing the visual analytics workspace, we conducted a literature review related to LM comparison tasks (e.g., [5,24,65]). Furthermore, we conducted two semi-structured interviews in an online setting with two NLP researchers (co-authors of this paper) with expertise in language modeling tasks to discuss further common evaluation-related analysis aspects. Our goal was to gather specific linguistically motivated analysis tasks and research challenges for the evaluation of adapted LMs. In the following, we describe the gathered requirements through *Models and Data* and *Users and Tasks* [45].

3.1 Models and Data

The NLP research focuses not only on developing and adapting new models with better performance but also on understanding the linguistic properties the models implicitly capture. Probing classifiers [12,29,37] and adversarial testing [20,41,58] are the most common methods used in computational linguistics to understand such properties. The current research explores not only what the models learn but also when they fail and which limitations they have, such as different types of biases [4,17,43]; as well as ways to mitigate those biases [14,16,56,57,72]. Visualizations are used to analyze the model latent spaces to gain insights into the degree of changes in embedding vectors [15,61], properties encoded in embedding vectors [62], and word neighborhood changes [5,24,65]. Especially, the comparison of embedding local neighborhoods is one of the critical tasks for many users of LMs [5,65]. For such comparisons, one first needs to select words for the analysis. Boggust et al. [5] write that this is commonly done either in a data- or model-driven way, for instance, by exploring specific domain-related words or challenging words for the analyzed model. During the interviews, the NLP researchers agreed with this statement and emphasized that evaluation methods related to model limitations often explore specific, pre-defined human-interpretable concepts such as *gender-related stereotypes*. When analyzing such human-interpretable concepts, people commonly analyze contextualized word embeddings. For some methods (e.g., Word Embedding Association Tests [7]), researchers compute word-level vectors without an explicit context [34,71]. In particular, for BERT, one can append the sequence start and the separator token before and after the word, respectively (e.g., [CLS] word [SEP]) and extract embeddings with context size zero [74] (also known as *decontextualized* embeddings [6]). In the following, we call them context-0 embeddings. Our experts also emphasized the need to 'connect' the embedding space with the model's behavior to inspect whether specific embedding vectors influence the model's predictions on downstream tasks.

3.2 Users and Tasks

With this work, we aim to support developers and researchers who adapt and evaluate LMs to perform their analysis more easily by focusing on the analysis of diverse human-interpretable concepts. To do that, we gathered task-related requirements. NLP researchers' work is related to comparison (i.e., baseline) tasks. In particular, their analysis typically involves **(T0)** a comparison of multiple LMs with different architectures or fine-tuning settings as well as multiple model layers. Second, they typically analyze specific human-interpretable concepts and try to **(T1)** partition the representation (e.g., embedding) space according to these concepts. Third, they try to **(T2)** understand interactions between specific concepts, e.g., to what extent these concepts are represented similarly in the representation (e.g., embedding) space. They aim to **(T3)** detect 'unexpected' associations, e.g., positive sentiment words that tend to trigger the negative sentiment because, e.g., they are negated. And finally, their goal is to **(T4)** connect the representation space with the actual behavior of the model, e.g., to understand whether concepts are separated in the representation space yet do not affect the behavior of the model.

4 Visual Analytics Workspace: Data Processing

In this section, we present our visual analytics workspace and its three main components: **Adapter Composition View** (in Fig. 2 A), **Explanation Composition View** (in Fig. 2 B), and **Visual Comparison View** (in Fig. 2 Workspace) for model and layer comparison. Before introducing the workspace design in Sect. 5, we describe the data processing.

4.1 Data Modeling

Motivated by the gathered requirements, we first build the data model. Since human-interpretable concept analysis plays a crucial role in NLP research, we start by modeling such concepts. By default, we work with concepts that are commonly used in research related to bias mitigation[3] and sentiment analysis. The users can upload further concepts as .json files in the interface. One concept is represented by two word lists, each having a specific polarity. For instance, a concept called *person names* consists of two word lists – *male person names* and *female person names*, respectively. We provide the following concepts: *male/female person names*, *male/female pronouns*, *male/female-related nouns*, *male/female-related stereotypes*, *positive/negative human qualities*, *high/low-GDP countries*, and words related to *weak/strong*, *family/career*, *science/arts*, *intelligence/appearance*.

We first model each word in a concept through a list of sentences in

[3] https://github.com/cisnlp/bias-in-nlp

which the word is used. For this purpose we use the Yelp dataset [73]; the user can also upload other datasets and use them for explanations. The associated sentences are used for two purposes. First, we use them as an input to the (adapted) LM to extract the word's contextualized word embeddings. The embeddings are extracted layerwise (i.e., layer 1-12 for BERT-base) and get aggregated [6] for each unique word (e.g., one average embedding from all occurrences of the word *Germany* per layer). Second, we use these sentences as input for task adapters for prediction making. Furthermore, we extract the word's context-0 embedding by using the model's special tokens and the word itself as the input to the model (i.e., [CLS] word [SEP]). For words that do not occur in the vocabulary, we average their sub-token embeddings.

4.2 Adapter Composition and Explanation Composition

We load adapters from AdapterHub repository and list them in the **Adapter Composition View**. The user can select an adapter for the analysis by clicking on the particular icon. Currently, we have pre-processed the data for six models: the **pre-trained** BERT (BERT-base-uncased), the **debiasing** BERT [34], and four task adapters for BERT (sentiment classifiers **sst-2**, **rotten-tomates** [54], and **imdb** [54], and the named entity recognizer **conll2003**). For a new adapter selection, the data is first pre-processed and stored in the database.

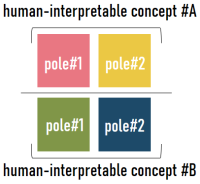
The user defines which explanation methods to use for their analysis in the **Explanation Composition View**. The explanations are constructed from available concepts and three visualization types. The visualizations include *Concept Embedding Similarity*, *Concept Embedding Projection*, and *Concept Prediction Similarity*. The Concept Embedding Similarity requires an input of two concepts: one is used as an anchor in the visualization and the other is explained through the cosine similarity to the anchor. The Concept Embedding Projection requires an input of one or two concepts (to analyze a single concept or the relation between two (un)related concepts). The user can choose between multiple projection techniques: Principal Component Analysis (PCA) [30], Multidimensional Scaling (MDS) [33], t-Distributed Stochastic Neighbor Embedding (t-SNE) [68], and Uniform Manifold Approximation and Projection (UMAP) [42]. The Concept Prediction Similarity can be applied only on adapters with prediction heads (e.g., sentiment classifier). The explanation requires an input of one concept; the class labels are used as anchors in the visualization.

The pre-computed adapters, as well as created explanations, are displayed on top of the **Visual Comparison View**, represented through an icon and adapter's or explanation's name. The user first selects an explanation type, then an adapter that they would like to analyze. To guide the users toward interesting adapters for the analysis, we display a glyph underneath the adapter's icon. The glyph shows the overlap between the two concept word lists for the selected explanation. The overlap is determined using a similar algorithm to the class consistency [64] that is commonly used to select good scatterplot views for high-dimensional data. An example of these glyphs is shown in Fig. 2. The explanation visualization is displayed in the **Visual Comparison View** on a zoomable canvas; hence, one can display as many explanations on the canvas as needed. A draggable placeholder icon ⊗ marks the position where the next selected adapter visualization will be displayed on the screen.

5 Visual Analytics Workspace: Design Rationale

In the following, we describe the design rationale and the visual encoding for the designed explanation visualizations. Our workspace supports the exploration of a single model and the comparison of two models or two model layers **(T0)**. We apply diverse explanation methods (i.e., the similarity in the high-dimensional space, embedding projection, and explanation details) to detect and avoid potential artifacts generated by a single approach (e.g., projection artifacts). The design of the comparison visualizations was motivated by the design guidelines by Gleicher [19] that consider the comparative elements, challenges that may occur, strategies to overcome the challenges, and the design solutions.

Global Visual Encoding In all visualizations, we use the visual mark called point [9] (i.e., rectangle) to represent words. Hidden word labels are displayed by hovering over a word's rectangle. We use positional encoding [9] to partition the embedding space **(T1)**, detect concept intersections **(T2)**, and locate 'unexpected' associations **(T3)**. The position is used to show the similarity between words according to underlying features such as different types of word embedding vectors or prediction labels. We group words belonging to the same concept through an additional visual mark, i.e., area/contour. The contours are implemented using the d3-contour library[4] based on a two-dimensional kernel density estimation on the point clouds. The user can specify how many contour lines to display in the visualization by moving a slider. To support memorization and ease the readability, we use a global color encoding [9] for concepts. In particular, we use two diverging color pairs. One color pair represents the two word lists of a concept. The selection of the color pairs was not trivial since the colors had two objectives: the separability between two concepts and the separability between two word lists of one concept. The final decision was made as follows: we selected two warm colors (i.e., **pink** and **yellow**) representing one concept and two cold colors (i.e., **green** and **blue**) representing the other, as shown in the side figure. Further color alternatives are included in the supplementary material.


human-interpretable concept #A
pole#1
pole#2
pole#1
pole#2
human-interpretable concept #B

Visual Encoding for Single Model Visualizations By default, we display as many details as possible in the single visualizations but avoid label overplotting. An algorithm measures whether displaying a label would lead to overlap. The algorithm iterates through words in both word lists of a concept and measures the bounding box of each text element that gets added to the visualization. If the new element creates an overlap, it is hidden in the visualization.

Visual Encoding for Model Comparison Visualizations For effective model comparison, we use both the juxtaposition design (see [19]) and either the superposition for visualizations that have a positional anchor or explicit encoding for visualizations that lack the positional anchor (e.g., projection techniques). By default, we show the summary [19] of the two models to avoid datapoint overplotting. The summaries are created using the contour library; the source model is represented through its contour in the 2D space, and the target model is represented through its filled-out area. We use the scan sequentially [19] strategy to show exact word positions. The filter icons are explained in Sect. 5.1.

5.1 Concept Embedding Similarity

This explanation displays the cosine similarity between two concepts enabling to partition the embedding space **(T1)**, detect concept intersections **(T2)**, as well as locate 'unexpected' associations **(T3)**. In this representation, one concept is used as an anchor for explanation purposes. The other concept can be the same as the anchor (e.g., *human qualities* used twice in Fig. 3) or it may differ from the anchor (e.g., *person names* as a concept and *pronouns* as an anchor in Fig. 7). We measure the average cosine similarity between a word in the concept to words in each pole of the selected anchor. It helps to analyze different biases in the data, for instance, whether, e.g., *female pronouns* are more similar to specific *stereotype* words than *male pronouns*.

(1) Single Model Explanation – The two anchor word lists represent the two axes in the scatterplot visualization (e.g., *negative qualities* represent y-axis and *positive qualities* represent x-axis in Fig. 3). The average similarity values between a word in the concept to the anchors are used as coordinates in the 2D visualization. A word's (e.g., *cheerful* in Fig. 3) average similarity to the first anchor word list (e.g., *negative qualities*) specifies the word's y-position and the average similarity to the second anchor word list (e.g., *positive qualities*) specifies the word's x-position. To support the readability, we add a diagonal line to the visualization as a point of reference. If a word is more similar to the first word list, then it will be located on the left-hand-side of the diagonal; if a word is more similar to the second word list, then it will be located on the right-hand-side of the diagonal. Words that are equally similar to both word lists are located on the diagonal. By default, we display all

[4] https://github.com/d3/d3-contour

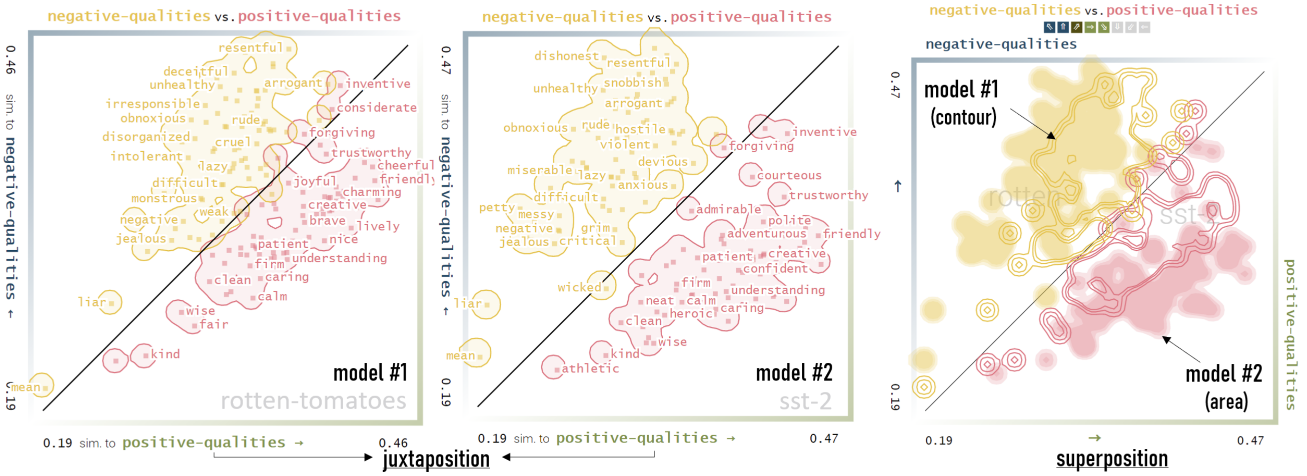


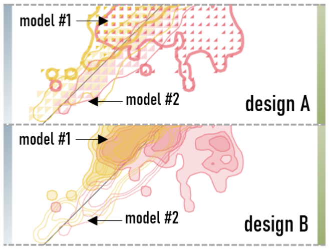
Fig. 3: We provide two types of model comparison designs for analyzing concept embedding **similarity**, i.e., **juxtapositon** where two models are displayed next to each other and **superposition**, where two models are displayed in one visualization. Here: the contextualized word embeddings extracted from layer 11 for the rotten-tomatoes and sst-2 sentiment classifiers differentiate between *positive-* and *negative human qualities*. The rotten-tomatoes model requires context to separate the two polarities since the separation is stronger than for context-0 embeddings (see Fig. 2).

words in the concept word lists as rectangles and show non-overlapping labels. Since most of the word lists consist of ca. 100 words, the visualization has overplotting issues that limit the analysis of concept intersections. To overcome these issues, we add a contour line around each pole. We use the d3-contours library and specify the bandwidth parameter to 5, which leads to larger areas for more dense regions; however, single outlier data points are enclosed in separate, smaller areas, enabling the detection of 'unexpected' associations **(T3)**. The area is colored in the particular concept's color with a decreased opacity.

(2) Model Comparison Explanation – As mentioned in Sect. 3, the overall goal of NLP researchers is to compare models or layers with respect to concept distributions **(T0)**. The design of comparison visualizations is not trivial, as described by Gleicher [19]. Thus, in order to consider all relevant aspects, we follow his design guidelines.

The comparison visualization for Concept Embedding Similarity has to display two models or layers simultaneously, each showing the distribution of concept words with respect to selected anchors. Two types of challenges may arise when designing for this objective: (1) the concepts, as well as models, may overlap, and (2) word similarity changes may produce patterns that are difficult to outline all at once. Before we describe the strategies to overcome these challenges, we name our design considerations. Gleicher [19] names three design alternatives for comparison visualizations: juxtaposition, superposition, and explicit encoding. In our workspace, each explanation can be explored in a juxtaposition design (shown in Fig. 3 left) since single model visualizations are always displayed next to each other on the screen. This representation has limitations, though. Since we use all the available 2D space for a single model to reduce word overlaps, the visualizations of the compared models often have different scales. Thus, the detailed model and concept overlap analysis is restricted. Therefore, instead of using juxtaposition, we place two models in the same representation using the superposition design (shown in Fig. 3, right). The superposition is a valid alternative since the Concept Embedding Similarity visualization has anchors (which is not the case for projection techniques, as described in the following).

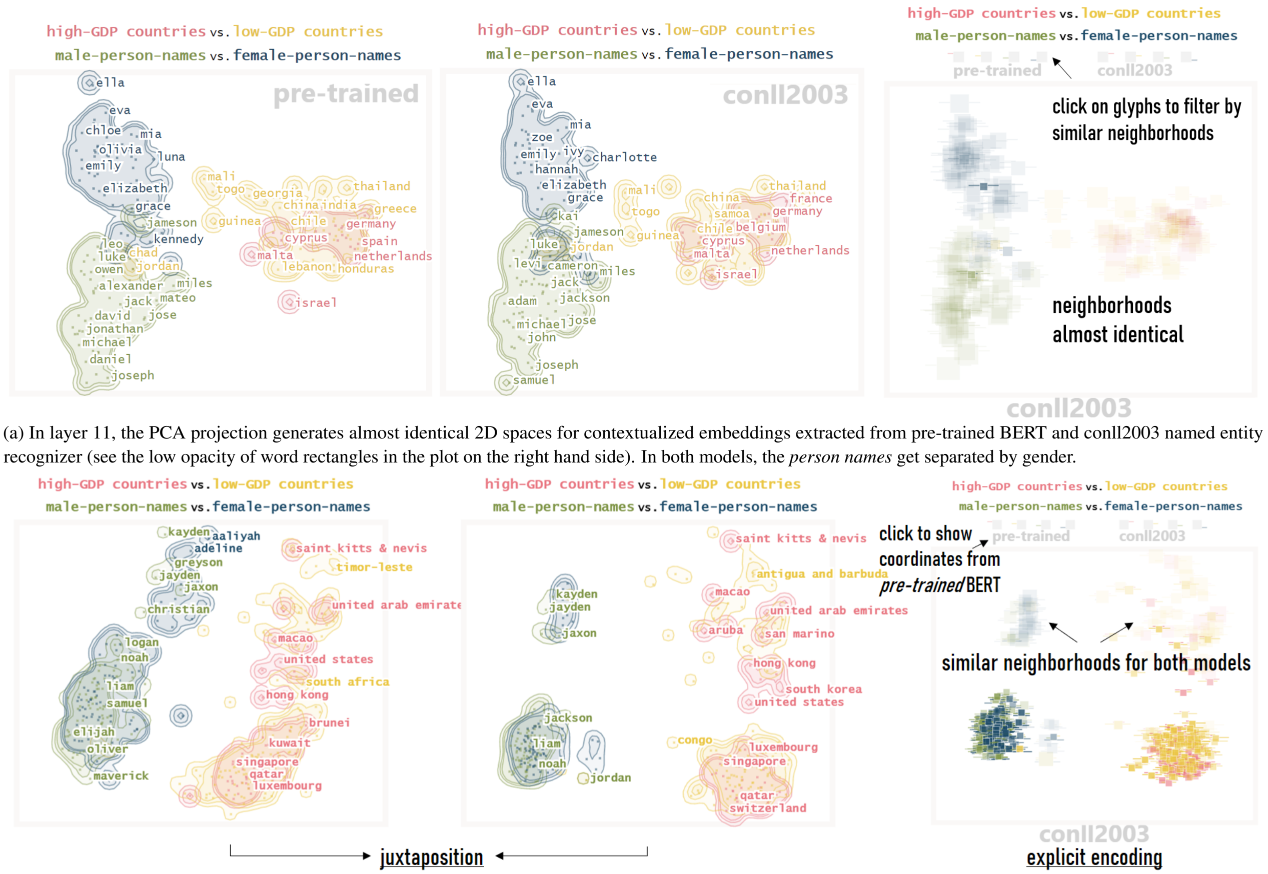
In the comparison visualization, we display the cosine similarity values between concept words and anchors for two models simultaneously **(T0)**. We follow the comparative visualization guidelines and apply two strategies that enable the analysis of overlapping concepts, models, and word similarity patterns. First, we provide a summary of the two models. We, therefore, display only the contours of their word positions; more details (e.g., word exact positions) are displayed on demand. During the design process, we created several alternative representations to visually separate the two models. Each designed alternative was discussed with a group of visual analytics experts to critically assess the representation's advantages and limitations. In particular, we created representations that showed two types of the density of the visualized words, i.e., discrete as well as continuous. The discrete representation displayed the density regions through triangles arranged on a grid layout, whereby each model was represented with triangles of different sizes and opacity (smaller rectangles with higher opacity for the target model, see design A in the side figure). The continuous representation summarized the models through their contours (see design B in the side figure). After several discussions, the latter was selected as the final design due to its visual smoothness and limited clutter. The final design is as follows: the first (i.e., source) model is displayed only through contour borders. Since the words themselves are not visible, we use multiple contour lines to highlight the density of the word-occurrence regions. The second (i.e., target) model is displayed through a filled-out area of the contour regions with transparency. In addition to the model summarization, we apply the scan sequentially strategy to enable the analysis of word similarity changes. For this purpose, we implemented filter buttons that can be used to highlight words that have common properties with respect to their positional changes (i.e., their position in the source model compared to their position in the target model in the 2D space). In particular, we measure the angle between the word's position in the source and the target model. By hovering over one of the filter buttons ⇖ ⇑ ⇗ ⇒ ⇘ ⇓ ⇙ ⇐, words with similar positional changes are highlighted in the visualization. The buttons themselves are colored according to the anchor to which words in the target model become more similar in comparison to the source model. An example of the word filtering is shown in Fig. 2.



5.2 Concept Embedding Projection

The second explanation method displays the words in a 2D visualization, whereby the 2D positions are obtained using a projection technique such as PCA on the embedding vectors. This explanation visually partitions the representation space **(T1)** and supports the analysis of concept intersections **(T2)**. Since in the Concept Embedding Similarity explanation we compute the similarity on high-dimensional vectors, this representation shows the similarity from a different modeling perspective.

(1) Single Model Explanation – The explanation displays words within one or two concepts, depending on whether the user wants to analyze one concept or the overlap of two (un)related concepts. Like in every visualization, we display words as rectangles and, by default, show labels for words that do not overlap. To support the readability of dense regions, we designed and discussed several design alternatives. First, we displayed words using a scatterplot technique, which is common for displaying projection data (design A

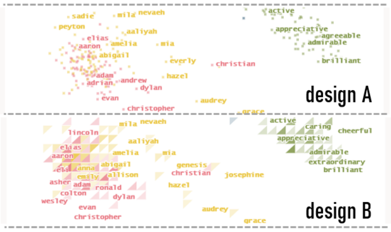


(a) In layer 11, the PCA projection generates almost identical 2D spaces for contextualized embeddings extracted from pre-trained BERT and conll2003 named entity recognizer (see the low opacity of word rectangles in the plot on the right hand side). In both models, the *person names* get separated by gender.

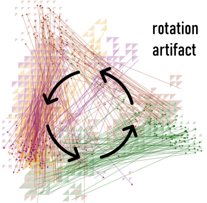
(b) In layer 11, the PCA projection of context-0 embeddings from conll2003 named entity recognizer produces four distinct clusters. Two clusters (with low opacity) have similar neighborhoods in both models. These are rare person names (e.g., Nevaeh) and long country names (e.g., Trinidad and Tobago). *Person names* do not encode gender.

Fig. 4: We provide two different types of model comparison designs for analyzing concept embedding **projections**, i.e., **juxtapositon** where two models are displayed next to each other and **explicit encoding** that summarizes embedding changes through word neighborhood overlaps.

in the side figure). Since the goal of the visualization is to clearly show concept intersections **(T2)**, however, words in the projection often overlap, this representation was not feasible. Second, we applied a kernel density estimation algorithm on the projected words to estimate and visualize the densest regions in the 2D space. We first represented the density through triangles displayed in a grid layout, whereas the density value was mapped to the triangles' opacity (design B in the side figure). Similar to the simple scatterplot, it was difficult to detect concept intersections easily. Thus, in the final design, we use multiple contours showing the estimated density of the different regions (Fig. 4). It allows detecting not only the densest regions but also words with unexpected associations **(T3)** (i.e., outliers).



(2) Model Comparison Explanation – Our goal is to display intersections and positional changes of one or two concept word lists. The challenge of this representation is grounded in the artifacts of the applied projection techniques. In particular, since we rely on projection techniques to compute word coordinates, the visualization lacks an interpretable point of reference; projection techniques typically come with artifacts such as rotation or flipping of the representation space, making the comparison of two spaces difficult. Like in all other visualizations, the user can explore model differences in a juxtaposition design since the single model explanations are always placed next to each other on the screen (as shown in Fig. 4b, left). The juxtaposition has limitations, though. If the compared models produce different embedding spaces (which is the case for most of the model and layer comparisons), they produce 2D spaces that are difficult to align. The insufficiency of the superposition design is depicted in the side figure. There, we represent a word's positional changes through lines, whereas a line connects the word's position in the source model with the position in the target model. Due to rotation artifacts, the comparison of word changes is restricted even if the changes are minor. Thus, for projection comparison purposes, we apply the third design alternative, i.e., the explicit encoding design (as shown in Fig. 4b, right).



For the explicit encoding, we first define relationships to encode in the visualization [19], i.e., we explain the projection changes through word nearest neighbors in the 2D space. In particular, after computing the projection's coordinates, we compute ten nearest neighbors for each word and store them as attributes in the data structure. When the user explores two models according to their embedding projections, we visually explain the neighborhood overlaps. This, according to design guidelines [19], is an example of the summarize strategy. Unlike the Concept Embedding Similarity visualization, we display only a single word's instance in the visualization. Its 2D coordinates, by default, are coordinates from the source model. The user can change it by clicking on the model's name in the visualization (shown in Fig. 4b, right). The neighborhood changes are displayed as follows. For each word, we measure the neighborhood overlap (the number of equal neighbors in the source and target model) and map it to the size of the word's rectangle representation. The higher the overlap, the larger the rectangle and the lower the opacity. Moreover, we add horizontal lines to the rectangle, each showing the nearest neighbors from the particular concept's pole. As shown in the side figure, in the pre-trained BERT the person-name *Maverick* is more similar to *countries* (**blue** and **green** lines on the left-hand-side) than *person names*; in the conll2003 named entity recognizer, this word

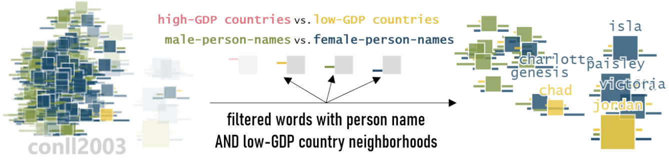


Fig. 5: Words with similar neighborhoods can be filtered by selecting particular glyphs. In conll2003 named entity recognizer, country names *Jordan* and *Chad* are more similar to *person names* than *countries*.

becomes more similar to *person names* (yellow and pink lines on the right-hand-side of the rectangle). An example of two models with similar word neighborhoods is shown in Fig. 4a and with different word neighborhoods – in Fig. 4b. If the word neighborhoods change, then rectangles are smaller with a higher opacity, as shown in Fig. 4b. In addition to the summarize strategy, we support the scan sequentially strategy to enable the analysis of word neighborhood changes. The users can filter words based on their neighborhoods by clicking on the glyph representations displayed on top of the visualization. The filtered words are highlighted; the rest are faded out (shown in Fig. 5). On mouse over a word, its nearest neighbors in the source model are highlighted; on click, the nearest neighbors in the target model are highlighted, enabling a simple neighborhood comparison.

5.3 Concept Prediction Similarity

The third visualization can be used on adapters that have been trained on two-class classification tasks. It explains the prediction similarity of two models that are trained on the same task, e.g., whether two sentiment classifiers produce similar prediction outcomes, and connects the representation space and the model's behavior **(T4)**. For this task, the user has to select one concept; the model then predicts class labels for the words' assigned sentences.

(1) Single Model Explanation – To provide an overview of prediction similarity, we aggregate the label information for all sentences in which the word is used in the corpus and use the average prediction to determine the word's x-coordinate in the visualization. In particular, we divide the number of sentences having the first prediction label (e.g., NEGATIVE sentiment) by the total number of sentences for the particular word; the more predictions with the first class label – the closer the point is to the beginning of the x-axis. If the predictions are equal for both class labels, the word is placed in the middle of the x-axis. The y-coordinate is determined by the word's position in the particular word list. The words themselves are displayed as rectangles.

(2) Model Comparison Explanation – In the comparison visualization, our goal is to show the prediction differences between two models **(T0)**. Since in this visualization we have clear anchors (the prediction labels), we can apply a similar design approach as for the Concept Embedding Similarity plot. In particular, we use both juxtaposition as well as superposition designs. In the superposition design, both models are represented in the same visualization, as shown in Fig. 6. We stick to the same design as for the Concept Embedding Similarity plot and first summarize the model predictions through contours. The source model is represented through the contour's borders; the target model's contours are filled out with a decreased opacity. The user can click on the filtering icons displayed on top of the visualization; the prediction changes are highlighted accordingly, supporting the scan sequentially strategy.

5.4 Explanation Details

When explaining model changes, researchers usually try to find the reasons for particular patterns in the data. Thus, we designed three visualizations to explain patterns in the comparison visualizations.

Context Concordance View – The patterns in the Concept Embedding Similarity visualization can be influenced by the word contexts (sentences) from which the contextualized word embeddings are extracted. Thus, for this visualization, we added a *Context Concordance View* that lists all sentences in which a word is used in the corpus (shown in Fig. 1, right). The view is displayed when clicking on the particular word in the Concept Embedding Similarity visualization. There, the selected word is highlighted for a better comparison.

Projection Artifact View – We propose a dense pixel visualization to explore the latent space and reveal semantically similar embeddings. The pixel visualization is inspired by Shin et al. [63] stripe-based visualization of word embeddings. The primary goal is to create a compact visual summary of the embeddings with all dimensions without using dimensionality reduction methods (e.g., PCA). The pixel visualization displays each embedding as a vertical pixel bar, a grid-shaped column where each colored pixel (rectangle) is an embedding feature value. Herefore, we normalize the embeddings to the unit length and color the pixels according to a diverging color scheme. Then we place the pixel bars next to each other on the x-axis, producing a dense pixel visualization. The y-axis displays the 768 embedding dimensions, and the rows are ordered by the median of the visualized embedding dimensions to highlight block and band patterns [2]. The x-axis can be reordered by linking and brushing in the single model explanations to interactively create clusters to highlight and display as a block of embeddings. Alternatively, the embeddings can be clustered using HDBSCAN [8] using cosine similarity to detect clusters of similar embeddings. We can explore clusters in latent space through clustering without relying on dimensionality reduction methods, which typically produce some artifacts. Overall, comparing the colored pixel bars enables us to perceive pairwise similarities between the embeddings and generate new insights into the latent space, such as identifying groups of similar embeddings, meaningful embedding dimensions, or outliers.

Prediction View – To explore the exact prediction differences in the Concept Prediction Similarity comparison visualization, we display the predicted labels for all sentences assigned to a word in the *Prediction View* (shown in Fig. 1, right). The view is displayed when selecting a word in the Concept Prediction Similarity visualization.

6 Evaluation

We conducted expert case studies [60] with the experts from the requirement analysis (see Sect. 3) to assess initial feedback on the visualization sufficiency for model comparison tasks. We further gathered positive (informal) feedback from two computational linguistic professors on the designed workspace. We present insights created for three out of six models introduced in Sect. 4.2: the pre-trained BERT, the debiasing adapter for BERT by Lauscher et al. [34], and the conll2003 named entity recognizer. We plan to extend the study with more participants to quantitatively evaluate the usability of the interface.

6.1 Expert Study Setup

The following insights were created collaboratively with two experts in natural language processing tasks. The study was conducted online in the form of a video conference. The experts had two main tasks: (1) to investigate models related to bias and (2) to explore the limitations of a named entity recognition model. The experts further analyzed predictions for sentiment classifiers **(T4)** as described in Sect. 5.3; however, they are not included in the case study description below due to the paper's space considerations. The study was concluded with a semi-structured interview about the workspace's usability.

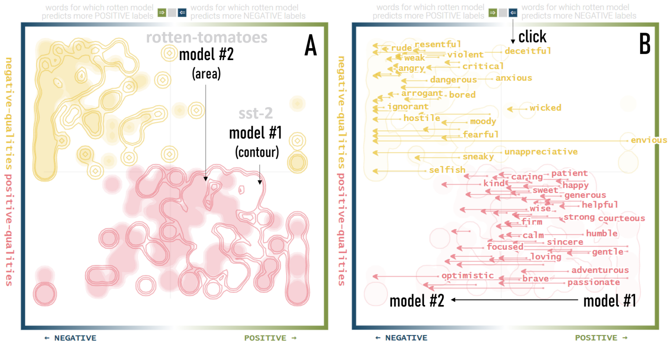


Fig. 6: Concept Prediction Similarity shows two sentiment classifiers (see A). Compared to the sst-2 model (contour borders), the rotten-tomatoes model (filled areas) classifies sentences with occurrences of *positive* and *negative human qualities* more often as **NEGATIVE** (B).

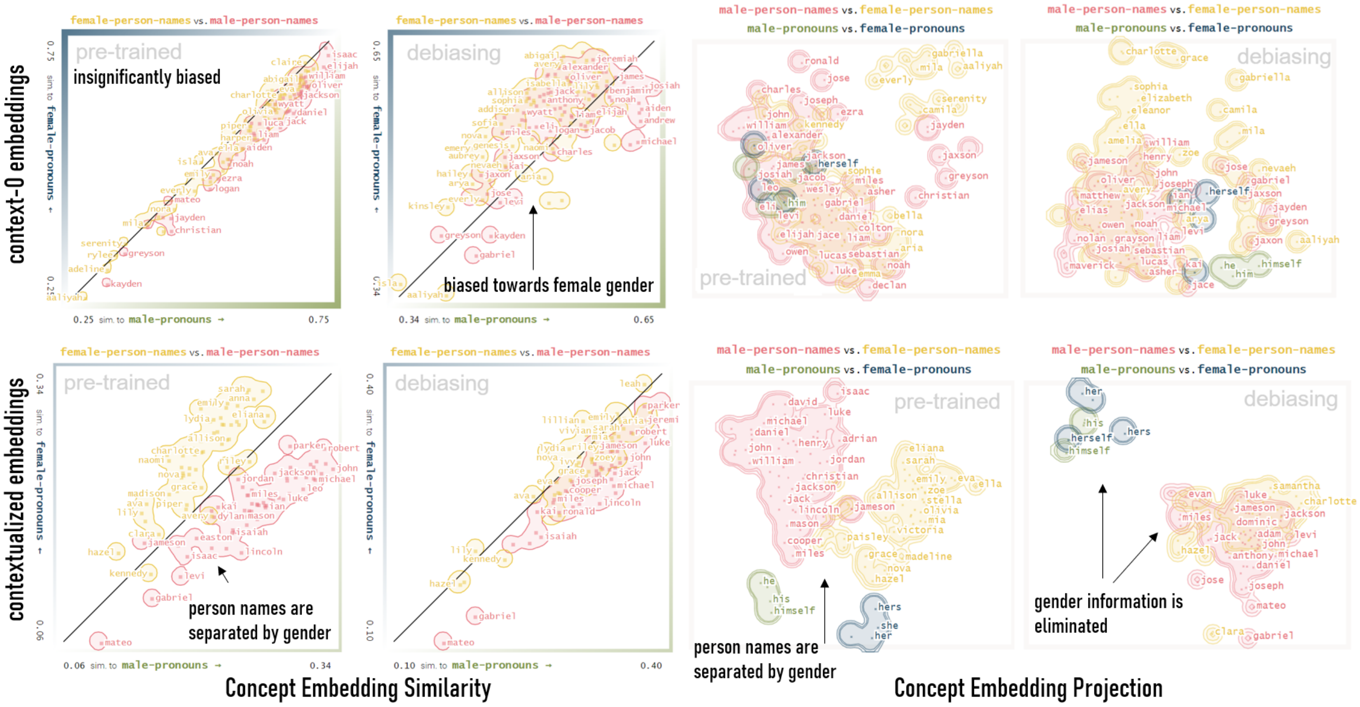


Fig. 7: Context-0 embeddings are used for evaluation purposes in Word Embedding Association Tests [34, 71]. Their produced spaces differ from the contextualized ones, though. Although context-0 embeddings suggest that the debiasing adapter by [34] inverts the gender bias of the pre-trained BERT, the PCA projection on contextualized embeddings shows that the adapter successfully eliminates the gender information.

Data – The data for the study included the 10 human-interpretable concepts introduced in Sect. 4.1. The contextualized word embedding representations were extracted from the Yelp dataset [73], whereby each word in the concept list was represented by up to 300 contexts.

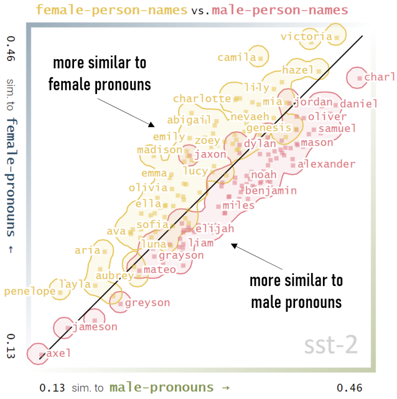
Tasks – For the analysis related to bias detection, the interface provides the debiasing model trained by Lausher et al. [35]. We use their evaluation results as ground truth to investigate whether the insights can be replicated using our workspace. In particular, the authors show that the model is effective in attenuating gender biases according to most of the applied evaluation methods. However, the results of the Word Embedding Association Test (WEAT) [7] are less successful. The WEAT test measures the association between two target word sets (e.g., *male pronouns*) and (e.g., *female pronouns*) based on their mean cosine similarity to words from two attribute sets (e.g., *science terms*) and (e.g., *art terms*) that is measured on context-0 (i.e., static [35]) word embeddings. Lauscher et al. observe that according to the WEAT test, the pre-trained BERT model is insignificantly biased; however, the debiasing adapter does not reduce the bias but instead – inverts it. The participants thus received the task to evaluate the particular adapter regarding two specific analysis tasks: (1) to inspect how the embedding space is partitioned for gender-related concepts **(T1)** and (2) to explore gender-related concept intersections **(T2)**.

Their second task was to analyze the conll2003 named entity recognizer concerning its learning capabilities of specific named entity categories such as *person names* and *countries*. Their particular analysis tasks were to investigate whether the model partitions the embedding space according to the different categories **(T1)**, whether there are intersections between the categories **(T2)**, and whether the model produces 'unexpected' associations **(T3)** between specific named entities.

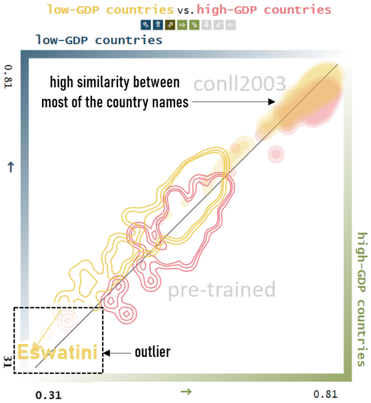
6.2 Expert Case Studies

In the following, we describe gained insights for the specified tasks.

(Task 1) Bias in Language Models – To gain insights into the gender-related concept representation and their intersections, the participants investigated the Concept Embedding Similarity visualization. They selected the pre-trained BERT and debiasing models and analyzed the word similarities between different concepts (e.g., *person names* as shown in Fig. 7) to *pronouns* that were displayed as anchors in the visualization. The visualization revealed that in the upper layers (e.g., layer 11) of the pre-trained BERT, context-0 embeddings for *person names* are slightly more similar to *male pronouns* than *female pronouns*, but the difference is insignificant. However, in debiasing adapter, most of these *person names* (even *male person names*) are more similar to *female pronouns*. Similar patterns could be observed for other concepts (e.g., *gender-related stereotypes*, *countries*), which matches the observations by Lauscher et al. [34]. It is important to notice that this 'bias inversion' is visible only for context-0 embeddings. When exploring the relationships between the same concepts computed on contextualized word embeddings (in Fig. 7), both Concept Embedding Similarity and Concept Embedding Projection visualizations show that the debiasing adapter was able to eliminate the gender information – the visualizations show no separation between the *person-name* and *pronoun* concepts. However, in the pre-trained BERT, *female person names* are more similar to *female pronouns* and *male person names* are more similar to *male pronouns*. The visualizations reveal that most of the models obtain the gender information from the word's context, and it is not encoded in the word (e.g., *person name*) itself. The only exception is the sst-2 sentiment classifier; there, even context-0 embeddings get separated by gender (side figure). Different to other adapters, the sst-2 model is trained on phrases extracted from Stanford parse trees rather than full sentences. Thus, words in isolation that are used to extract the context-0 embeddings present an unnatural input to most of the models [6]; however, the input is less unnatural for the sst-2 model since some of its training instances are one or two words long.



(Task 2) Named Entity Recognition – To analyze the learning capabilities of the conll2003 named entity recognizer, the participants explored the Concept Embedding Similarity visualization for the concept *low/high-GDP countries* – two word lists, each grouping countries with a similar GDP rank according to 2020 statistics. As shown in the side figure, the conll2003 model learns that most of the countries are similar without encoding their welfare (see the top-right corner). By exploring the word positions, one can see that the model does not recognize the country *Eswatini* since its similarity to both *low-GDP* and *high-GDP countries* is low (0.31) in comparison to other countries that have a similarity of circa 0.8.



Next, the participants analyzed the model's distinction between *person names* and *country names* – a typical task for a named entity recognizer. The Concept Embedding Projection visualization of the two concepts is shown in Fig. 4. In the early layers, both models produce similar word neighborhoods and the *person names* and *country names* have a poor separation. In upper layers (e.g., layer 11 in Fig. 4b), the projection of conll2003 embeddings displays four clusters. One cluster contains country names (Fig. 8 cluster A) and another – person names (Fig. 8 cluster B). The neighborhoods of the two smaller clusters are similar to those in the pre-trained BERT, suggesting that the *conll2003* model did not capture any new properties for these particular words. By interactively exploring the word neighborhoods, one can observe that one cluster consists of rare person names (e.g., *Nevaeh*), whereas the other contains relatively long country names (e.g., *Trinidad and Tobago*). Since the visualizations show the context-0 embeddings, the *person names* are not separated by gender. To investigate whether the four clusters are artifacts generated by the PCA projection, the embeddings values were displayed in the Projection Artifact View. Fig. 8 shows that the values for embedding vectors within one cluster produce similar patterns, suggesting that the four clusters are not the projection's generated artifacts. The separation between long and short *country names*, as well as common and rare *person names*, might be a reason of long and rare words not being in the BERT's vocabulary; thus, this might be an artifact of averaging sub-token embedding vectors and must be further investigated.

6.3 Preliminary Expert Feedback

The experts provided positive feedback concerning the workspace's applicability for model evaluation and comparison tasks. They described the interface to be intuitive and easy to use. The experts found it useful having the option to choose between different concepts, and in particular–with respect to bias–different ways to quantify it. This allows them to evaluate the models along 'different axes', and this is in accordance with works that have shown that bias is manifested in multiple ways. The experts also appreciated the ability to analyze both the representations and the predictions that provide two complementary ways to explain a model: the prediction-based view focuses on the more high level 'interface' (i.e., model's predictions) while the representation analysis focuses on its actual working mechanism (i.e., how these predictions are derived). The workspace also demonstrates and makes use of one of the advantages of adapters over other fine-tuning methods – the fact they are easily integrated into one pre-trained model without having to fine-tune a different model per task.

One important advantage of our workspace was described by the experts as follows. Adapters are usually tested in-domain (e.g., people train for the sentiment task and evaluate on sentiment prediction). The 'side-effects' the training has on other aspects are often unaddressed. Thus, it was appreciated that the workspace puts emphasis on evaluating a given adapter according to metrics that are not necessarily related to the main tasks it was trained on. The interface with its diverse concepts brings another advantage, particularly for the bias evaluation tasks. According to the experts, while certain notions of bias are well studied, the more interesting cases are those which are more subtle and less intuitive or straightforward. The workspace makes it easier to explore the representation space of the models and potentially discover new notions of bias, or more generally, undesired properties of the model in question, as depicted in the Sect. 6.2. The limitations of the workspace are formulated as research opportunities in the following section.

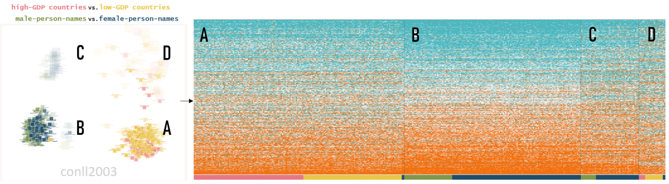


Fig. 8: In *Projection Artifact View*, the user can explore embedding vectors aligned as columns in a pixel visualization. We use a bipolar color scale to show vector values (from min **blue** to max **orange**).

7 Discussion and Research Opportunities

In the previous section, we presented how we can use our workspace to gain insights into model specificities. During the design and evaluation process, we discovered several opportunities for future research.

Comparison of Numerous Models – Currently, our workspace supports the direct comparison of two models at a time. An interesting research challenge would be to display more than two models in the same comparison visualization. While designing our visualizations, we faced challenges in how to select designs that allow visually separate the two models. By displaying more than two models simultaneously, one would need to come up with new visual design alternatives.

Supporting Model Fine-Tuning – Our work is a step toward effectively comparing adapter models. It is still limited to explorative tasks and, at this point, does not actively suggest which actions to undertake to improve the adapter performances. We see, however, this as a very important direction for future work. The system should provide insights into the models' strengths and limitations and, in an ideal case, also provide hints or suggestions on which steps should be overtaken (e.g., adaptation of the training dataset) to improve the models' performances.

Visual Explanations Combined with Probing Classifiers – During our collaboration, the NLP researchers mentioned several potential extensions concerning the functionality of the workspace. Since they commonly train classifiers to investigate concept intersections, they mentioned this as an extension to the visual explanation methods. The two methods used in parallel could increase their trust in the generated insights. In particular, if the projection and the classifier produce similar results, it is more likely to be true and less likely to be an artifact of the particular method in use.

Support for Adapter Training – Currently, our workspace supports the analysis of adapters from the AdapterHub repository. The framework, however, supports different adapter composition techniques, such as adapter stacking [50] as well as their fusion [48]. We plan to extend the workspace in a way that researchers could train new adapters in the interface by applying the different adapter composition methods and directly evaluate their created representation spaces, which, hopefully, would lead to better-performing models for downstream tasks.

8 Conclusion

We presented a novel visual analytics workspace for the analysis and comparison of LMs that are adapted for different masked language modeling and downstream classification tasks. The design was motivated by requirements gathered during a literature review and collaboration with NLP researchers. We introduced three new comparison visualizations: Concept Embedding Similarity, Concept Embedding Projection, and Concept Prediction Similarity that were designed by applying the comparative visualization guidelines by Gleicher [19]. We show the applicability of the workspace through expert case studies, confirm findings from the related work, and generate new insights into adapter learning properties. A demo is available as part of the LingVis framework [13] under: https://adapters.demo.lingvis.io/.

Acknowledgments

This paper was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) within projects BU 1806/10-2 "Questions Visualized" of the FOR2111, and the ETH AI Center.

REFERENCES

[1] D. L. Arendt, N. Nur, Z. Huang, G. Fair, and W. Dou. Parallel Embeddings: A Visualization Technique for Contrasting Learned Representations. In *Proc. of the 25th Int. Conf. on Intelligent User Interfaces*, pp. 259–274, 2020.

[2] M. Behrisch, B. Bach, N. Henry Riche, T. Schreck, and J.-D. Fekete. Matrix reordering methods for table and network visualization. In *Computer Graphics Forum*, vol. 35, pp. 693–716. Wiley Online Library, 2016.

[3] M. Berger. Visually Analyzing Contextualized Embeddings. In *IEEE Visualization Conf. (VIS)*, pp. 276–280. IEEE Computer Society, Los Alamitos, CA, USA, oct 2020.

[4] S. L. Blodgett, S. Barocas, H. Daumé III, and H. Wallach. Language (technology) is power: A critical survey of "bias" in NLP. In *Proc. of the Association for Computational Linguistics*, pp. 5454–5476. Association for Computational Linguistics, Online, July 2020.

[5] A. Boggust, B. Carter, and A. Satyanarayan. Embedding Comparator: Visualizing Differences in Global Structure and Local Neighborhoods via Small Multiples. In *27th Int. Conf. on Intelligent User Interfaces*, pp. 746–766, 2022.

[6] R. Bommasani, K. Davis, and C. Cardie. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4758–4781. Association for Computational Linguistics, Online, July 2020. doi: 10.18653/v1/2020.acl-main.431

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

[8] R. J. Campello, D. Moulavi, and J. Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, pp. 160–172. Springer, 2013.

[9] M. S. T. Carpendale. Considering visual variables as a basis for information visualisation. PRISM, 2003.

[10] J. F. DeRose, J. Wang, and M. Berger. Attention flows: Analyzing and comparing attention mechanisms in language models. *IEEE Trans. on Visualization and Computer Graphics*, 2020.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[12] D. Edmiston. A Systematic Analysis of Morphological Content in BERT Models for Multiple Languages. *arXiv preprint arXiv:2004.03032*, 2020.

[13] M. El-Assady, W. Jentner, F. Sperrle, R. Sevastjanova, A. Hautli, M. Butt, and D. Keim. lingvis.io – A Linguistic Visual Analytics Framework. In *Proc. of the Association for Computational Linguistics: System Demonstrations*, pp. 13–18, 2019.

[14] Y. Elazar and Y. Goldberg. Adversarial removal of demographic attributes from text data. In *Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing*, pp. 11–21, 2018.

[15] K. Ethayarajh. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In *Proc. of the Conf. on Empirical Methods in Natural Language Proc. and the Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP)*, pp. 55–65. ACL, Hong Kong, China, Nov. 2019.

[16] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Int. Conf. on Machine Learning*, pp. 1180–1189. PMLR, 2015.

[17] I. Garrido-Muñoz, A. Montejo-Ráez, F. Martínez-Santiago, and L. A. Ureña-López. A survey on bias in deep nlp. *Applied Sciences*, 11(7):3184, 2021.

[18] G. Glavaš, A. Ganesh, and S. Somasundaran. Training and domain adaptation for supervised text segmentation. In *Proc. of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 110–116. Association for Computational Linguistics, Online, Apr. 2021.

[19] M. Gleicher. Considerations for visualizing comparison. *IEEE Trans. on Visualization and Computer Graphics*, 24:413–423, 2018.

[20] M. Glockner, V. Shwartz, and Y. Goldberg. Breaking NLI Systems with Sentences that Require Simple Lexical Inferences. In *Proc. of the 56th Association for Computational Linguistics*, pp. 650–655. ACL, 2018.

[21] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proc. of the Association for Computational Linguistics*, pp. 8342–8360. Association for Computational Linguistics, Online, July 2020.

[22] X. Han and J. Eisenstein. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *EMNLP*, 2019.

[23] F. Heimerl and M. Gleicher. Interactive analysis of word vector embeddings. In *Computer Graphics Forum*, vol. 37, pp. 253–265. Wiley Online Library, 2018.

[24] F. Heimerl, C. Kralj, T. Moller, and M. Gleicher. embcomp: Visual interactive comparison of vector embeddings. *IEEE Trans. on Visualization and Computer Graphics*, 2020.

[25] B. Hoover, H. Strobelt, and S. Gehrmann. exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformers Models. In *Proc. of the Association for Computational Linguistics, System Demonstrations*. ACL, 2020.

[26] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *Int. Conf. on Machine Learning*, pp. 2790–2799. PMLR, 2019.

[27] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. In *Proc. of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339. Association for Computational Linguistics, Melbourne, Australia, July 2018.

[28] S. Jain and B. C. Wallace. Attention is not explanation. In *NAACL*, 2019.

[29] G. Jawahar, B. Sagot, and D. Seddah. What does BERT learn about the structure of language? In *Proc. of the Association for Computational Linguistics*, pp. 3651–3657. ACL, Florence, Italy, July 2019.

[30] I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Philosophical Trans. of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[31] Y. Kim, P. Petrov, P. Petrushkov, S. Khadivi, and H. Ney. Pivot-based transfer learning for neural machine translation between non-English languages. In *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP)*, pp. 866–876. Association for Computational Linguistics, Hong Kong, China, Nov. 2019. doi: 10.18653/v1/D19-1080

[32] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proc. of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[33] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[34] A. Lauscher, T. Lueken, and G. Glavaš. Sustainable modular debiasing of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 4782–4797. Association for Computational Linguistics, Punta Cana, Dominican Republic, Nov. 2021.

[35] A. Lauscher, O. Majewska, L. F. R. Ribeiro, I. Gurevych, N. Rozanov, and G. Glavaš. Common Sense or World Knowledge? Investigating Adapter-Based Knowledge Injection into Pretrained Transformers. In *Proc. of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pp. 43–49. Association for Computational Linguistics, Online, Nov. 2020.

[36] Q. Li, K. S. Njotoprawiro, H. Haleem, Q. Chen, C. Yi, and X. Ma. Embeddingvis: A visual analytics approach to comparative network embedding inspection. In *2018 IEEE Conf. on Visual Analytics Science and Technology (VAST)*, pp. 48–59. IEEE, 2018.

[37] Y. Lin, Y. C. Tan, and R. Frank. Open Sesame: Getting inside BERT's Linguistic Knowledge. In *Proc. of the ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 241–253. ACL, Florence, Italy, Aug. 2019.

[38] Z. Lin, A. Madotto, and P. Fung. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 441–459. Association for Computational Linguistics, Online, Nov. 2020.

[39] S. Liu, P.-T. Bremer, J. J. Thiagarajan, V. Srikumar, B. Wang, Y. Livnat, and V. Pascucci. Visual exploration of semantic relationships in neural word embeddings. *IEEE Trans. on Visualization and Computer Graphics*, 24(1):553–562, 2017.

[40] S. Liu, Z. Li, T. Li, V. Srikumar, V. Pascucci, and P.-T. Bremer. Nlize: A perturbation-driven visual interrogation tool for analyzing and interpreting natural language inference models. *IEEE Trans. on Visualization and Computer Graphics*, 25(1):651–660, 2018.

[41] R. Marvin and T. Linzen. Targeted Syntactic Evaluation of Language Models. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing*, pp. 1192–1202. ACL, Brussels, Belgium, Oct.-Nov. 2018.

[42] L. McInnes, J. Healy, N. Saul, and L. Grossberger. UMAP: Uniform Manifold Approximation and Projection. *The Journal of Open Source Software*, 3(29):861, 2018.

[43] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.

[44] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[45] S. Miksch and W. Aigner. A matter of time: Applying a data–users–tasks design triangle to visual analytics of time-oriented data. *Computers & Graphics*, 38:286–290, 2014.

[46] C. Park, I. Na, Y. Jo, S. Shin, J. Yoo, B. C. Kwon, J. Zhao, H. Noh, Y. Lee, and J. Choo. Sanvis: Visual analytics for understanding self-attention networks. In *IEEE Visualization Conf. (VIS)*, pp. 146–150. IEEE, 2019.

[47] J. Pennington, R. Socher, and C. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543. Association for Computational Linguistics, Doha, Qatar, Oct. 2014. doi: 10.3115/v1/D14-1162

[48] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. pp. 487–503, 2021.

[49] J. Pfeiffer, A. Rücklé, C. Poth, A. Kamath, I. Vulić, S. Ruder, K. Cho, and I. Gurevych. AdapterHub: A framework for adapting transformers. In *Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pp. 46–54. Association for Computational Linguistics, Online, 2020.

[50] J. Pfeiffer, I. Vulić, I. Gurevych, and S. Ruder. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7654–7673. Association for Computational Linguistics, Online, Nov. 2020.

[51] M. Q. Pham, J. M. Crego, F. Yvon, and J. Senellart. A study of residual adapters for multi-domain neural machine translation. In *Proc. of the Fifth Conf. on Machine Translation*, pp. 617–628. Association for Computational Linguistics, Online, Nov. 2020.

[52] J. Phang, T. Févry, and S. R. Bowman. Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. *ArXiv*, abs/1811.01088, 2018.

[53] J. Philip, A. Berard, M. Gallé, and L. Besacier. Monolingual adapters for zero-shot neural machine translation. In *Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4465–4470. Association for Computational Linguistics, Online, Nov. 2020.

[54] C. Poth, J. Pfeiffer, A. R"uckl'e, and I. Gurevych. What to pre-train on? Efficient intermediate task selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10585–10605. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, Nov. 2021.

[55] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897, 2020.

[56] S. Ravfogel, Y. Elazar, H. Gonen, M. Twiton, and Y. Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proc. of the Association for Computational Linguistics*, pp. 7237–7256, 2020.

[57] S. Ravfogel, M. Twiton, Y. Goldberg, and R. D. Cotterell. Linear adversarial concept erasure. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, eds., *Proc. of the 39th Int. Conf. on Machine Learning*, vol. 162 of *Proc. of Machine Learning Research*, pp. 18400–18421. PMLR, 17–23 Jul 2022.

[58] K. Richardson, H. Hu, L. S. Moss, and A. Sabharwal. Probing Natural Language Inference Models through Semantic Fragments. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pp. 8713–8721. AAAI Press, 2020.

[59] A. Rogers, O. Kovaleva, and A. Rumshisky. A Primer in BERTology: What We Know About How BERT Works. *Trans. of the Association for Computational Linguistics*, 8:842–866, 2020.

[60] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Trans. on Visualization and Computer Graphics*, 18(12):2431–2440, Dec. 2012.

[61] R. Sevastjanova, A.-L. Kalouli, C. Beck, H. Hauptmann, and M. El-Assady. Explaining Contextualization in Language Models using Visual Analytics. In *Proc. of the Association for Computational Linguistics*, ACL. ACL, 2021.

[62] R. Sevastjanova, A.-L. Kalouli, C. Beck, H. Hauptmann, and M. El-Assady. LMFingerprints: Visual Explanations of Language Model Embedding Spaces through Layerwise Contextualization Scores. *Computer Graphics Forum*, 41(3):295–307, 2022.

[63] J. Shin, A. Madotto, and P. Fung. Interpreting word embeddings with eigenvector analysis. In *32nd Conf. on Neural Information Processing Systems (NIPS 2018), IRASL workshop*, 2018.

[64] M. Sips, B. Neubert, J. P. Lewis, and P. Hanrahan. Selecting good views of high-dimensional data using class consistency. In *Computer Graphics Forum*, vol. 28, pp. 831–838. Wiley Online Library, 2009.

[65] V. Sivaraman, Y. Wu, and A. Perer. Emblaze: Illuminating machine learning representations through interactive comparison of embedding spaces. In *27th Int. Conf. on Intelligent User Interfaces*, pp. 418–432, 2022.

[66] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. S eq 2s eq-v is: A visual debugging tool for sequence-to-sequence models. *IEEE Trans. on Visualization and Computer Graphics*, 25(1):353–363, 2018.

[67] H. Strobelt, B. Hoover, A. Satyanaryan, and S. Gehrmann. LMdiff: A visual diff tool to compare language models. In *Proc. of the 2021 Conf. on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 96–105. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, Nov. 2021.

[68] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11), 2008.

[69] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proc. of the 31st Int. Conf. on Neural Information Processing Systems*, NIPS'17, p. 6000–6010. Curran Associates Inc., Red Hook, NY, USA, 2017.

[70] J. Vig. A Multiscale Visualization of Attention in the Transformer Model. In *Proc. of the Association for Computational Linguistics: System Demonstrations*, pp. 37–42. Association for Computational Linguistics, Florence, Italy, July 2019.

[71] I. Vulić, S. Baker, E. M. Ponti, U. Petti, I. Leviant, K. Wing, O. Majewska, E. Bar, M. Malone, T. Poibeau, R. Reichart, and A. Korhonen. Multi-SimLex: A Large-Scale Evaluation of Multilingual and Crosslingual Lexical Semantic Similarity. *Computational Linguistics*, 46(4):847–897, 02 2020. doi: 10.1162/coli_a_00391

[72] Q. Xie, Z. Dai, Y. Du, E. Hovy, and G. Neubig. Controllable invariance through adversarial feature learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[73] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds., *Advances in Neural Information Processing Systems*, vol. 28. Curran Associates, Inc., 2015.

[74] M. Zhao, P. Dufter, Y. Yaghoobzadeh, and H. Schütze. Quantifying the Contextualization of Word Representations with Semantic Class Probing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1219–1234. Association for Computational Linguistics, Online, Nov. 2020.