Journal of Biomedical Informatics 100S (2019) 100057

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

A survey of word embeddings for clinical text ☆

Faiza Khan Khattak [a,c,d], Serena Jeblee [a,c], Chloé Pou-Prom [a,d], Mohamed Abdalla [a,c], Christopher Meaney [b,c], Frank Rudzicz [a,c,d,e,*]

[a] Department of Computer Science, University of Toronto, Toronto, Ontario, Canada
[b] Department of Biostatistics, University of Toronto, Toronto, Ontario, Canada
[c] Vector Institute for Artificial Intelligence, Toronto, Ontario, Canada
[d] Li Ka Shing Knowledge Institute, St Michael's Hospital, Toronto, Ontario, Canada
[e] Surgical Safety Technologies Inc, Toronto, Ontario, Canada

ARTICLE INFO

Keywords:
Word embeddings
Clinical data
Natural language processing

ABSTRACT

Representing words as numerical vectors based on the contexts in which they appear has become the *de facto* method of analyzing text with machine learning. In this paper, we provide a guide for training these representations on clinical text data, using a survey of relevant research. Specifically, we discuss different types of word representations, clinical text corpora, available pre-trained clinical word vector embeddings, intrinsic and extrinsic evaluation, applications, and limitations of these approaches. This work can be used as a blueprint for clinicians and healthcare workers who may want to incorporate clinical text features in their own models and applications.

1. Introduction

With the increasing adoption of electronic health records (EHRs) in clinical settings, jurisdictions and organizations involved in the provision of healthcare are amassing large volumes of data on the patients they serve. Many EHR implementations permit healthcare providers to record free-text clinical notes on their patients. These notes are rich in clinical and demographic information, describing: current problems, past medical history, family history, treatment and immunization history, referrals or consults, as well as overall progress towards health goals. It is becoming increasingly common for healthcare organizations to adopt methods from machine learning (ML) and artificial intelligence (AI) to leverage clinical text data for task automation, clinical predictive modelling, as well as knowledge discovery and understanding. Relevant information can be gleaned from free-text clinical notes. However, free-text clinical notes are unstructured, riddled with spelling errors, and consist of a language very specific to the medical domain [1]. Converting the free-text in clinical notes into a representation that can easily be used by ML remains one of the top challenges in healthcare.

In this paper, we discuss a popular method for representing the semantics of text data – word embeddings – and focus on how this method can be meaningfully applied to represent clinical text data. Specifically, we describe model training methods (including details on the data required to learn these representations), evaluation procedures, and applications to clinical data.

A word embedding is a real-valued vector that represents a single word based on the context in which it appears. This numerical word representation allows us to map each word in a vocabulary to a point in a vector space, as exemplified by Fig. 1. The 'distributional hypothesis' states that words that occur in the same contexts have similar or related meanings [2]. Thus, we expect that the embeddings for semantically or syntactically related words will be closer to each other than to unrelated words in vector space. This relatedness is entirely dependent on the text data, or corpus, from which these embeddings are derived.

Historically, feature engineering in natural language processing (NLP) involved creating specific numerical functions to represent salient aspects of the text, such as the ratio of nouns to pronouns. This approach often required significant domain knowledge and effort to identify meaningful features. By contrast, word embeddings can be learned directly from a corpus of text and do not require any manual labeling or feature extraction/engineering, i.e., they can be learned in an *unsupervised* manner. As such, word embeddings can be easily learned on any corpus of text data (including clinical text from EHRs).

A basic recipe for training, evaluating, and applying word embeddings is presented in Fig. 2. Section 2 describes different word embed-

☆ This article was originally published in Journal of Biomedical Informatics: X. Journal of Biomedical Informatics: X is now discontinued and the article is republished here for the reader's convenience. For citation purposes, please use the publication details of this article: Journal of Biomedical Informatics, 100S.

* Corresponding author at: Department of Computer Science, University of Toronto, Toronto, Ontario, Canada.
E-mail addresses: faizakk@cs.toronto.edu (F.K. Khattak), sjeblee@cs.toronto.edu (S. Jeblee), poupromc@smh.ca (C. Pou-Prom), mohamed.abdalla@mail.utoronto.ca (M. Abdalla), christopher.meaney@utoronto.ca (C. Meaney), frank@cs.toronto.edu (F. Rudzicz).

https://doi.org/10.1016/j.yjbinx.2019.100057
Available online 28 October 2019

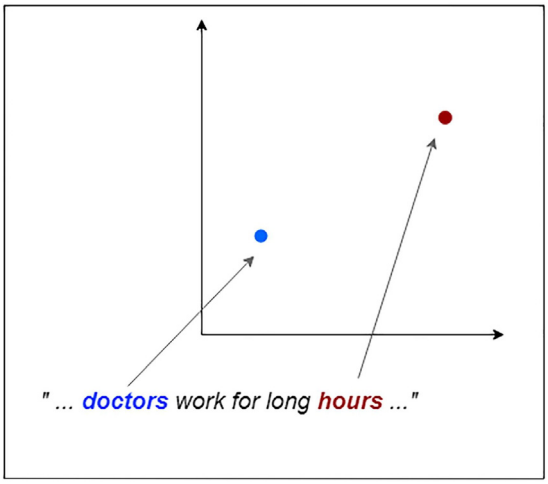

Fig. 1. Visual example of word embeddings. Each word in a sentence gets mapped to a *word embedding*. For simplicity, we only show the embeddings as points in 2-D space.

ding types, with a particular focus on representations commonly used in healthcare text data. We give examples of corpora typically used to train word embeddings in the clinical context, and describe pre-processing techniques required to obtain representative word embeddings in Section 3. We then focus on practical aspects of training these word embeddings in Section 4, and discuss model fitting and hyper-parameter tuning. We devote Section 5 to the evaluation of word embeddings, including both *intrinsic* and *extrinsic* methods. Finally, Section 6 addresses various limitations of word embeddings, such as *bias*, *interpretability*, and *privacy*.

2. Representing text with embeddings

In this section, we briefly describe different types of word embeddings, summarized in Table 1 (with additional details provided in the Appendix). Throughout this section, we refer to a sequence of n words in a given text as an 'n-gram'.

2.1. Word embedding representations

There are many different methods for learning word embeddings from a corpus, often beginning with a *one-hot encoding*, which maps each word in a vocabulary, consisting of V unique elements, to a unique index in a vector. A word is therefore represented by a vector of all zeroes except for a 1 in the appropriate position. By learning context-based prediction, word embedding methods map each of these one-hot vectors to dense representations, whose dimensionality is typically much lower than the size of the vocabulary, and whose elements capture the latent semantics of the language data. The rationale is that better word prediction is possible by learning better dense representations of words.

2.1.1. Word2vec

Word2vec is a prediction-based method that can be implemented in two ways: as a continuous bag-of-words (CBOW) and as a skip-gram (SG) [3–5]. Both the CBOW and SG models use small neural networks to learn the mapping of words to a point in a vector space, as detailed in [6]. The difference between these methods is in whether the neural network attempts to predict a focus word given its context (CBOW) or the reverse. Two key parameters for training word2vec embeddings are 1) the number of the embedding dimensions (typically between 50 and 500, tuned experimentally), and 2) the length of the context window (i.e., how many words before and after the target word should be used as context for training the word embeddings, usually 5 or 10 words). Other meaningful hyper-parameters are discussed in the Appendix. While training embeddings with more dimensions typically requires more training data, each dimension should capture some aspect of meaning, so the embeddings need to be large enough to differentiate words.

Doc2vec and paragraph2vec [7] are variants of word2vec in which vectors represent documents or paragraphs instead of words, respectively. Doc2vec can further be differentiated into two types – the paragraph vector-distributed memory (PV-DM) model (analogous to word2vec's SG model) and the paragraph vector-distributed bag of words (PV-DBOW) (similar to CBOW).

2.1.2. Global Vectors (GloVe)

The GloVe model also learns word embeddings, but from a term co-occurrence matrix instead of a word prediction task [8]. A co-occurrence matrix is a $V \times V$ matrix where V is the vocabulary size. Each entry of the matrix corresponds to the number of times the indicated vocabulary items occur together within a pre-specified context window, which moves across the entire corpus. GloVe learns vector embeddings so as to minimize the reconstruction error between co-occurrence statistics predicted by the model and global co-occurrence statistics observed in the training corpus. The model consists of numerous hyper-parameters that must be judiciously chosen, including the vector embedding dimension and the context window size.

Word vectors estimated using GloVe are conceptually similar to those derived from word2vec but uses an underlying count-based model, rather than word2vec's prediction-based model. Because GloVe typically computes statistics over larger context windows than word2-vec, it permits capturing longer-term dependencies, although the *order* of those dependencies will be lost. Empirically, no clear advantage has emerged for either word2vec or GloVe, as the overall performance depends on various factors, including: the type of data and evaluation task being considered.

2.1.3. FastText

The *FastText* model builds on a specific limitation of methods such as word2vec and GloVe. Specifically, it can handle new, out-of-vocabulary (OOV) terms by extending the word2vec skip-gram (SG) model with internal sub-word information [9], in the form of character n-grams (i.e., sequences of adjacent characters). The method builds a vector representation for a word based on the composition of these subword components, which allows the model to represent the morphology and lexical similarity of words, in addition to being able to construct vectors for unseen words.

Probabilistic FastText [10] combines Gaussian mixture models (GMMs) and FastText. Each word is represented as a GMM of K components, representing K different senses of a word. This representation is able to capture the sub-word structure, different word senses, and can provide better representation of rare or unseen words [10].

2.1.4. Embedding from Language Models (ELMo)

ELMo [11] is a contextualized word- and character-level embedding. Instead of using a fixed embedding for each word, ELMo looks at the entire sentence as it assigns each word an embedding. It uses a bi-directional recurrent neural network (RNN) trained on a specific task to create the embeddings. Since it uses a bidirectional architecture, the embedding is based on both the next and previous words in the sentence.

A key innovation of ELMo is that the embedding one obtains is weighted by a coefficient for the *task*, so that the same architecture may be trained on one task (e.g., movie reviews from social media) and then adapted on a very different task (e.g., diagnostic code prediction from EMRs), and combine information that is shared while also focusing on more specific semantics.

2.1.5. Bidirectional Encoder Representations from Transformers (BERT)

BERT [12] is another contextualized word representation model based on a multilayer bi-directional *transformer-encoder*, where the transformer neural network uses parallel attention layers rather than sequential recurrence [13]. BERT is pre-trained on two unsupervised tasks: (1) a 'masked language model, where 15% of the tokens are randomly masked (i.e., replaced with the "[MASK]" token), and the model

is trained to predict the masked tokens, (2) a 'next sentence prediction' (NSP) task, where the model is given a pair of sentences and is trained to identify when the second one follows the first. This second task is meant to capture more long-term or pragmatic information.

BERT is trained on the BooksCorpus dataset (800 M words) [14] and text passages of English Wikipedia. Two pre-trained model sizes for BERT are available: BERT-Base and BERT-Large. BERT can be used directly from the pre-trained model on un-annotated data, or fine-tuned on one's task-specific data. The pre-trained publicly available model and code for fine-tuning are available online[1,2]. Many domain-specific versions of BERT are available, which are trained or fine-tuned on in-domain text, including the following:

- **BioBERT** [15] is initialized with the general BERT model and pre-trained on PubMed abstracts and PMC full-text articles. It is further fine-tuned for biomedical text mining tasks such as named entity recognition (NER), question answering, and relation extraction. The pre-trained BioBERT model[3] and code[4] are publicly available.
- **ClinicalBERT** [16,17] is trained on clinical text from approximately 2 M notes in the MIMIC-III database [18], a publicly available dataset of clinical notes. More specifically, the following versions are presented by Alsentzer et al.[16]:
  1. *Clinical BERT*: Initialized from BERT-Base, uses text from all note types.
  2. *Clinical BioBERT*: Initialized from BioBERT, uses text from all note types.
  3. *Discharge Summary BERT*: Initialized from BERT-Base, uses only discharge summaries.
  4. *Discharge Summary BioBERT*: Initialized from BioBERT, uses only discharge summaries.

The code and pretrained models are available[5].

Huang et al.[17] also developed ClinicalBERT. ClinicalBERT uses the same pre-training tasks as the original BERT (i.e., masked token prediction and next sentence prediction). ClinicalBERT is pre-trained on a random sample of 100,000 notes from MIMIC-III [18] using the same parameter setting as the BERT-Base model. After pre-training, the model is fine-tuned on a task specific to clinical data. Code for training this version of ClinicalBERT is publicly available[6] as well as model parameter checkpoints[7].

- **SciBERT** [19] is trained on a random sample of 1.14 M full-text papers from Semantic Scholar [20] (18% computer science papers, 82% biomedical papers). There are four versions of SciBERT (code and pretrained models are publicly available[8]):
  1. Cased (vocabulary contains both uppercase and lowercase).
  2. Uncased (vocabulary converted to lowercase).
  3. Models using BaseVocab (original vocabulary released with BERT-Base) fine-tuned on BERT-Base models.
  4. Models using SciVocab (scientific vocabulary built using SentencePiece[9] on their scientific corpus), trained from scratch.

2.2. Other text embedding representations

Most word embeddings are represented in the Euclidean space, which sometimes makes them unable to capture hierarchical structure observed in certain corpora (or they may require high dimensional embedding dimensions in order to capture this complexity). *Poincaré embeddings* [21] avoid this problem by representing the data in the hyperbolic space (or more precisely, on a Poincaré ball). Unlike Euclidean space, hyperbolic space has a constant negative curvature and the ability of preserving the hierarchical/tree-like structure between points, which helps to capture the hierarchy and similarity of terms more accurately.

The main advantage of these embeddings is that they are purported to capture hierarchical information in far fewer dimensions than typical word embedding methods (see the Appendix for more details).

Word2vec, GloVe, and FastText learn models of language through windows of context, but this does not permit for long-term dependencies to be learned. For example, if a treatment plan at the end of a long paragraph is related to symptoms mentioned at the beginning, those methods may not be able to capture it. Secondly, often ambiguous words can only be disambiguated by using information much later in the text. These two insights led to a relatively state-of-the-art methodology, called *embeddings from language models (ELMo)* [11]. This approach employs a type of neural network, called a *recurrent neural network* (RNN) in which information is retained from one word to the next within the latent space of the embedding (see Fig. 3b). This latent space therefore encapsulates both information about the current word, but also to some extent all words that preceded it. In fact, ELMo uses a bi-directional RNN that encapsulates all preceding information as well as all information that is to follow, when learning the meaning of a given word (see Fig. 3c). This results in a representation that is highly contextual.

Some other embedding methods using a similar recurrent approach have been developed, including *Universal Language Model Fine-Tuning* (ULMFiT) [22], *Generative Pre-Training* (GPT) [23], *Generative Pre-Training 2* (GPT2) [24], XLNet [25], *Cross-linguage Language Model* (XLM) [26], and *Enhanced Language Representation with Informative Entities* (ERNIE) [27] but have not yet been used extensively in the medical domain.

2.3. Embedding visualization

Modern word embedding methods typically represent words in relatively high-dimensional spaces. To facilitate interpretation, it is often desirable to project these onto 2D or 3D surfaces for visualization.

A common method for embedding words into low dimensional spaces is t-distributed stochastic neighbor embedding (t-SNE) [28]. Visualizations can help illustrate the validity of learned embeddings; in particular, one expects to see semantically similar words cluster in these low-dimensional spaces (see Table 2). Fig. 5 shows a word embedding visualization using t-SNE on a model trained on the PubMed dataset[10]. Here, we can see that medical specialties are generally clustered together, as are body parts and the names of diseases; this is also demonstrated in Table 8.

3. Data and pre-processing

Learning word embeddings from clinical text requires data preparation, model fitting, and model evaluation.

3.1. Corpora

Embeddings have been generated from several varieties of clinical text corpora, as summarized in Table 3. Here, we present comparisons of learned embeddings across different corpora.

The utility of embeddings is heavily reliant on the task at hand. Wang et al. [29] compared embeddings generated from clinical notes, PubMed Central (PMC) (a repository of biomedical publications), Google News, and Wikipedia + Combined[11] (English newswire text data from a variety of tasks). They found that embeddings generated from clinical notes are more strongly correlated with human judgments of word similarity. These embeddings are also more useful in predicting fractures than embeddings trained on the other corpora. However, in other tasks such as smoking prediction, drug interaction extraction,

[1] https://github.com/google-research/bert.
[2] https://github.com/huggingface/pytorch-transformers.
[3] https://github.com/naver/biobert-pretrained.
[4] https://github.com/dmis-lab/biobert.
[5] https://github.com/EmilyAlsentzer/clinicalBERT.
[6] https://github.com/kexinhuang12345/clinicalBERT.
[7] http://bit.ly/clinicalbert_weights.
[8] https://github.com/allenai/scibert.
[9] https://github.com/google/sentencepiece.
[10] https://www.ncbi.nlm.nih.gov/pubmed/.
[11] https://catalog.ldc.upenn.edu/LDC2011T07.

and other information retrieval tasks, the differences due to corpora are not as significant.

Publicly-available pre-trained word embeddings may suffice for certain tasks, but typically training on text specific to the target domain leads to improved performance [41]. Publicly-available pre-trained clinical word embeddings[12] have been trained on PubMed abstracts[13], PMC full-text documents[13], PubMed abstracts and PMC full-text documents[13], and a combination of PubMed and PMC texts[13] along with texts extracted from Wikipedia[14]. MIMIC-III is a relatively large, freely available, relatively de-identified critical care dataset from approximately 40,000 patients. MIMIC-III contains free-text notes, laboratory tests, vital signs, and ICD-9 codes [18]. Additionally, the i2b2 (Informatics for Integrating Biology and the Bedside) center provides discharge summaries and clinical notes[15]. Although these datasets are smaller than general-purpose corpora, they include gold-standard annotations and have been used for shared tasks in biomedical NLP.

3.2. Pre-processing for word embeddings

Pre-processing a text prepares it for further processing. Typical steps include tokenization, removing stop words (e.g., *the*, *is*, *a*) and lemmatization (i.e., reducing different morphological forms of a word into a root form, e.g., both *running* and *runner* are lemmatized to *run*). Table 4 shows an example of a pre-processed sentence.

Care must be taken in pre-processing the clinical text. In particular, there is an increased risk that pre-processing medical acronyms may lead to ambiguity (e.g., the medical condition *ADD* (attention deficit disorder) may be converted to the verb *add* by pre-processing, if a blanket standardization to lowercase is applied).

4. Training word embeddings for clinical data

In this section, we discuss the process of training word embeddings and how different researchers have trained their own word embeddings on clinical text datasets.

4.1. Training

Training word embeddings involves fitting a model to a pre-processed corpus, and tuning the model's *hyper-parameters*, which are settings whose values are specified empirically before training. For example, the number of layers in a neural network is a hyper-parameter, whereas the weights between artificial neurons are parameters that are adjusted during the training phase.

Often, performance increases with the size of the dataset, up to a certain point. In experiments with medical data from PubMed abstracts, Zhu et al. [42] found that performance gains decrease after 4 million distinct words of training data. By contrast, Zhao et al. [43] found that training on a smaller, in-domain medical dataset provided better performance than training on a large, general domain dataset (Google News), based on the results of relatedness assessment, neighbourhood coherence, outlier detection, and drug name recognition. In a similar experiment, Wang et al. [29] showed that word embeddings trained on biomedical corpora (EMR data from the Mayo Clinic and MedLit articles from PubMed Central[16]) captured the semantics of medical terms better than those trained on general domain corpora (i.e., GloVe and Google News), but may not have better results for downstream biomedical NLP tasks such as biomedical information retrieval.

4.2. Existing clinical word embeddings

Many researchers train their own word embeddings on the clinical data available to them. However, there are also many large, freely available datasets that can be used for training word embeddings alone, or in combination with task-specific data in fairly arbitrary ways.

Dubois et al. [32] trained GloVe embeddings using 27 million anonymized notes obtained from 1.2 million patients and 49 million visits. When pre-processing the medical notes, they treated negative counterparts of words as a single word; e.g., *fever* and *no fever*, since negations are important in a patient's medical evaluations [44,45]. They also trained embeddings on journal abstracts (MCEMJ) from OHSUMED[17]. Huang et al. [46] trained word embeddings on MedHelp online forums (informal medical discussions), PubMed text (academic papers), and Wikipedia (general-purpose text), while De Vine et al. [30] trained word embeddings on concatenations of i2b2's training set [47], MedTrack [48], and CLEF's (Conference and Labs of the Evaluation Forum) 2013 data sets.

4.3. Embeddings for clinical concepts

Because word embeddings represent the meaning of a word based on the context in which it appears, they do not necessarily capture clinical knowledge. However, embeddings can be augmented to represent more than words – they can represent clinical concepts such as UMLS terms or ICD-10 codes. In order to improve the embedding representation, clinical concepts can be integrated into the model using several methods.

Starting with word embeddings, clinical knowledge can be used to refine the representations. This information can be injected at the training stage; for example, Boag et al. [49] injected domain knowledge into word embeddings trained on the MIMIC-III corpus [18] by adding UMLS features to the embedding training. Using an extension of word2vec that allows training on arbitrary contexts, Levy et al. [50] trained a model using a context representation that included both the surrounding words and the identifiers of UMLS terms that match that word in the UMLS database. This trained vector model is called Augmenting Word Embeddings with a Clinical Metathesaurus (AWE-CM)[18]. Although these embeddings had higher correlations with physician similarity judgments (.508 vs .495) on MiniMaySRS-doctors (see Section 5.1.1) [51], the default word2vec embeddings trained on MIMIC-III had higher correlation with medical coders (MiniMaySRS-coders and MayoSRS).

Similarly, Patel et al. [35] created embeddings for words and ICD-10 codes for automated medical coding review. They learned embeddings for the codes by treating the code as the target word and the terms in each document as the context. The adapted embeddings gave a 1% improvement of F_1 scores on an automated coding review task, where the model predicts whether the medical billing code should be accepted or sent for re-coding.

If word embeddings have already been trained, they can be adapted to the clinical domain using a technique called 'retrofitting', which adjusts the values of embeddings to better align with an external reference. Yu et al. [52] retrofitted word embeddings of clinical concepts from the Medical Subject Headings (MeSH) taxonomy using Faruqui et al.'s approach [53][19]. This retrofitting function essentially minimizes the distance between the original embedding and the retrofitted embedding for each word, while also minimizing the distance between embeddings of words that are similar in the MeSH hierarchy or according to the UMLS-similarity tool [54].

Zhao et al. [43] trained word embeddings (using word2vec skip-gram) for drug name entities from data extracted from PubMed and DrugBank [20], and tuned them according to correlation with word frequency vectors. The code and embeddings are available[21].

Embeddings can also be learned directly for medical terms. Choi et al. [55] trained embeddings for clinical concepts from UMLS, as well as ICD-9 codes. They used the OHSUMED dataset, as well as a private dataset of medical claims. The medical claims consisted of temporal data such as diagnosis and procedure codes, so in order the adapt

[12] http://bio.nlplab.org/.
[13] https://www.ncbi.nlm.nih.gov/pubmed/.
[14] https://en.wikipedia.org/wiki/Wikipedia:Database_download.
[15] https://www.i2b2.org/NLP/DataSets/Main.php.
[16] https://www.ncbi.nlm.nih.gov/pmc/.
[17] http://davis.wpi.edu/xmdv/datasets/ohsumed.
[18] https://github.com/wboag/awecm.
[19] https://github.com/mfaruqui/retrofitting.
[20] https://www.drugbank.ca/releases/latest.
[21] https://github.com/chop-dbhi/drug_word_embeddings.

the data to make it resemble text, they treated each 1/3 of a year as a pseudo-sentence, and randomly ordered the concepts within each segment. A bi-linear skip-gram embedding model was then trained using word2vec. The embeddings for UMLS identifiers and ICD-9 codes are also available[22].

Using a co-occurrence matrix of UMLS identifiers in clinical notes, the *cui2vec* model from Beam et al. [56] trained GloVe and word2vec embeddings [57]. The authors also developed a set of benchmarks to measure embedding similarity for co-morbidity relationships, causative relationships, drug-condition relationships, UMLS semantic types, and human similarity judgments from UMNSRS. Compared to the embeddings from Choi et al. [55], *cui2vec* performed better on almost all benchmarks. The embeddings are available[23].

However, these methods only looked at relationships between clinical concepts. In practice, we may want to consider the context in which those concepts appear (i.e., the rest of the words in the document); for example, we may need to handle negation and modality aspects (e.g., actual, hypothetical, conditional).

In order to use contextual information, Mencia et al. [58] used the all-in-text method [59], which creates label embeddings that are close in vector space to the embeddings of documents that have those labels. This is similar to the method from Patel et al. [35], described above, but here embeddings are trained from scratch. The authors jointly learned embeddings of words and documents, and manually assigned labels by minimizing the ranking loss over the similarity between document embeddings and their associated label embeddings (positive vs. negative). The learned embeddings showed higher correlation with the UMNSRS datasets [60] (see Section 5.1.1) than other pre-trained medical embeddings such as the PubMed vectors. The data are available online[24].

Zhu et al. [61] presented a framework for clinical concept extraction using contextual word embeddings. They trained ELMo word embeddings on SNOMED CT [62], in addition to discharge summaries and radiology reports from the MIMIC-III dataset [18].

These works consistently showed that training embeddings on medical text produced better downstream performance than training on larger, more general corpora such as Google News [58,43]. See Table 5 for a summary of available embeddings for clinical concepts.

4.4. Transforming target data to word embeddings

Once the word embeddings have been trained on the training data, they can readily be used to convert any target data to the word embeddings format. More specifically, it is not required to train word embeddings for the data of your choice each time, rather words from the target data can be mapped to word vectors using the trained word embeddings, replacing the OOV words with zeros (if not automatically handled by the word embedding algorithm). Then target data word embeddings can be used for an extrinsic evaluation (discussed in the next section) and/or a target task.

5. Evaluating clinical word embeddings

5.1. Intrinsic evaluation

From a linguistic perspective, trained word embeddings can be used to learn about semantic or syntactic relationships between words in a corpus. That is, the intrinsic statistics of a language within a corpus can be used to gain an understanding of the domain language itself.

5.1.1. Intrinsic evaluation methods

For general word embeddings, a typical intrinsic evaluation is to calculate similarity or relatedness scores to human judgments, using datasets such as SimLex-999 [63], WordSim353 [64], and MEN [65]. SimLex-999 contains similarity scores for 999 pairs of words generated from a human free-association test, WordSim353 contains relatedness judgments for 353 pairs of words in the general domain, and MEN contains relatedness judgements for 3000 word pairs from Wikipedia and the web.

However, good performance on these tests often does not correspond to good performance on downstream tasks such as classification[66]. This could be partially because many word similarity evaluation sets fail to distinguish between similarity and relatedness. For example, in medical text, *fever* and *cough* might be highly related, but they are distinct symptoms, and thus should be treated differently by the model. To address this issue, some evaluation datasets have separate subsets for testing similarity and relatedness, but others which are meant to test for 'similarity' actually conflate both similarity and relatedness into one concept, without distinguishing between the two. This is a problem that is present in both the WordSim353 and MEN datasets, while SimLex specifically measures similarity and not relatedness.

Another method of evaluation is by learning an alignment between the word embeddings and manually-constructed feature vectors with linguistics properties [67]. These linguistic vectors can be created from SemCor [68], a corpus of word lemmas annotated with coarse semantic categories from WordNet [69]. The alignment score is the sum of the correlations of aligned dimensions of the word embeddings and corresponding linguistic vectors. The subspace alignment scores show higher Pearson correlation ($r = 0.87$) with extrinsic evaluation scores such as Senti [70] (a treebank dataset for sentiment classification) than SimLex ($r = 0.51$) and WS-353 ($r = 0.46$). The evaluation code is available[25].

These standard measures do not necessarily test for the kind of semantic similarity that is important for medical data, described below. In order to evaluate embeddings for medical text, several domain-specific evaluation methods have been developed.

The UMNSRS similarity and relatedness datasets [60] contain pairs of clinical concepts from the Unified Medical Language System (UMLS) that were manually rated by medical experts. The similarity set contains 566 pairs of concepts and the relatedness set contains 588 pairs. These can be used as a reference to see if the cosine similarity scores of each pair according to a word embedding model are close to human judgments.

Similarly, the MayoSRS dataset [51] contains 101 clinical term pairs with relatedness scores from 3 physicians and 9 medical coders. With regards to the difference between similarity and relatedness judgments, the UMNRSRS dataset may be preferable because it explicitly encodes this distinction.

Choi et al. [55] introduced a Medical Conceptual Similarity Measure (MCSM) for UMLS concepts based on discounted cumulative gain (DCG), which a measure typically used for scoring information retrieval results. The function measures the similarity of a set of types by looking at the k nearest neighbors of each concept to see if they have the same type. They also introduced a similar Medical Relatedness Measure (MRM) based on the ICD-9 hierarchical groupings and relations between different types of clinical terms, which is defined in the appendix.

5.1.2. Intrinsic evaluation of word embeddings for clinical text

Chiu et al. [72] evaluated their word embeddings in both intrinsic (UMNSRS-Rel and UMNSRS-Sim) and extrinsic evaluation tasks (named entity recognition (NER) on BioCreative II Gene Mention task corpus (BC2) [73] and the JNLPBA corpus (PBA) [74]). They experimented with using the original text, a lower-cased version of the text, and by shuffling the sentences. They found contradicting results in intrinsic and extrinsic evaluation tasks. For example, increasing the context window parameter showed significant increases in the intrinsic evaluation, but a decrease in extrinsic measures. They also found that training on a larger corpus did not necessarily guarantee better word embeddings.

Wang et al. [29] evaluated word embeddings from four different sources: 1) 100-dimensional word vectors trained from the Mayo

[22] https://github.com/clinicalml/embeddings.
[23] http://cui2vec.dbmi.hms.harvard.edu/.
[24] http://www.ke.tu-darmstadt.de/resources/medsim.
[25] https://github.com/ytsvetko/qvec.

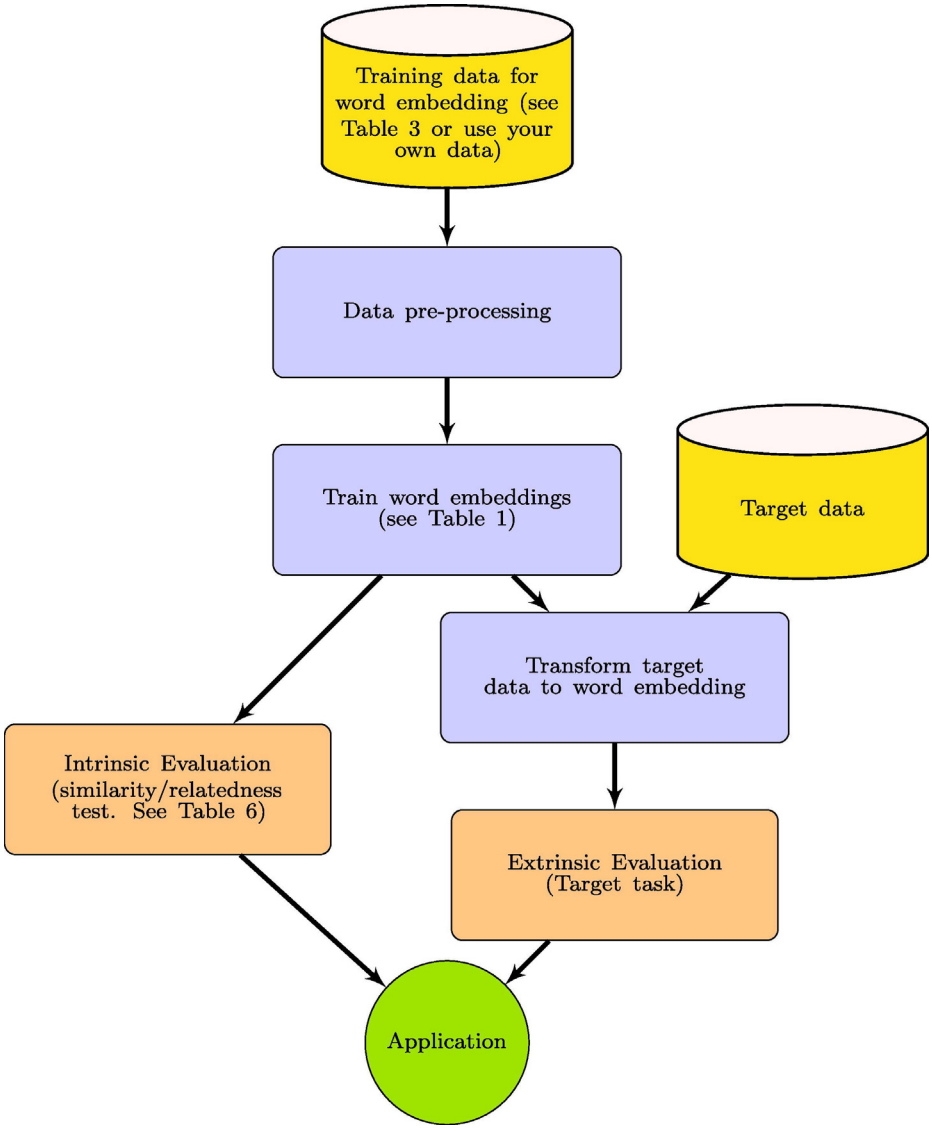

Fig. 2. How to train word embeddings on clinical data and use them for a machine learning task. If using a BERT or ELMo model, the transformation to word embeddings and target task (e.g., classification) are typically combined into a single step.

Clinic clinical notes, 2) 60-dimensional word vectors trained on the PubMed Central biomedical publications [41], 3) pre-trained publicly-available 100-dimensional from Wikipedia GloVe, and 4) standard 300-dimensional word vectors from Google News. They first performed a qualitative analysis by looking at the five most similar words to a given set of medical words, and found that word embeddings trained from Mayo Clinic notes and PMC biomedical publications captured medical words better than GloVe and Google News. Intrinsic evaluation consisted of evaluating the semantic similarities of medical terms on four biomedical measurement datasets: (1) Pedersen [51], (2) Hliaoutakis [75], (3) MayoSRS [76], and (4) UMNSRS [56,58]. Similarity was categorized as *practically synonymous*, *related*, *marginally related*, and *unrelated*. FastText computed word vectors for out-of-vocabulary words. Intrinsic evaluation revealed that similarity scores based on word embeddings trained on clinical notes were closer to similarity scores provided by humans, by calculating correlation coefficients.

Huang et al. [46] defined an elementary vector representations (EVR) as an n-dimensional binary vector $\boldsymbol{v}$, where $v_w[i] = 1$ if word w and the word at the i^{th} position occur within a context window (of length 5) in the text, $v_w[i] = 0$ elsewhere. They compared EVRs and word embeddings trained on MedHelp online forums, PubMed text (academic papers), and Wikipedia.

Table 1
Comparison of word embeddings model characteristics, where V is vocabulary size, and D is an arbitrary positive number. V is typically 1000 or $10,000$, while D usually varies from 50 to 500.

Model	Freq. based	Prediction based	Handles OOV	Sub-word info	Order of dim.
One-hot Encoding					V
Co-Occurrence Matrix	✓				V^2
CBOW		✓			D
Skip-gram		✓			D
GloVe	✓				D
FastText		✓	✓	✓	D
Poincaré Embedding		✓			< 10
ELMo		✓	✓	✓	D
Probabilistic FastText		✓	✓	✓	D
BERT		✓	✓	✓	D

To analyze these word embeddings, they considered medical terms from two sources: UMLS and Ranker[26] (an online social platform to perform polls on aspects such as entertainment, brands, sports and culture).

[26] https://www.ranker.com/.

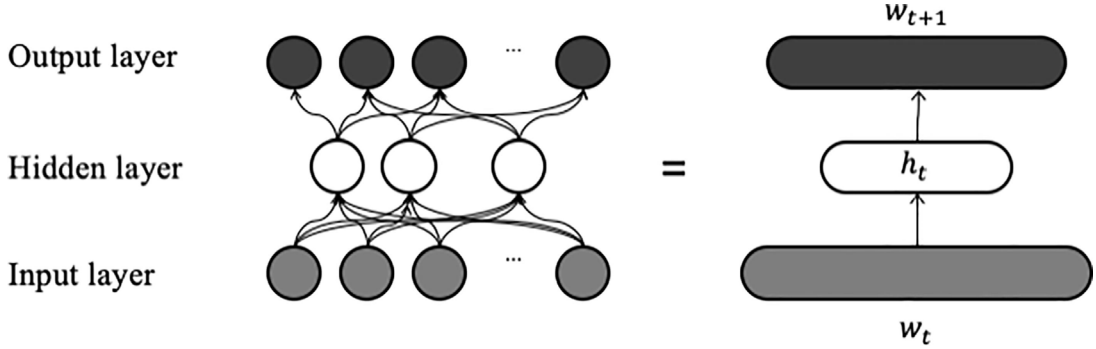

Fig. 3. (a) NN, (b) RNN, (c) ELMo. Three forms of neural network language model, each of which take input word w_t at time t (dark grey layer) and try to predict the next word (i.e., output w_{t+1}, the black layer) by learning some intermediate representation (i.e., the hidden layer (white), h_t). Figure (a) shows two equivalent representations of traditional, feed-forward neural networks; Figure (b) shows a recurrent neural network in which the hidden (latent) information is remembered forward over time; and Figure (c) shows the ELMo language model in which information is remembered forward *and* backwards in time to disambiguate words (where the backwards embedding is shown in hashed layers, h'_t).

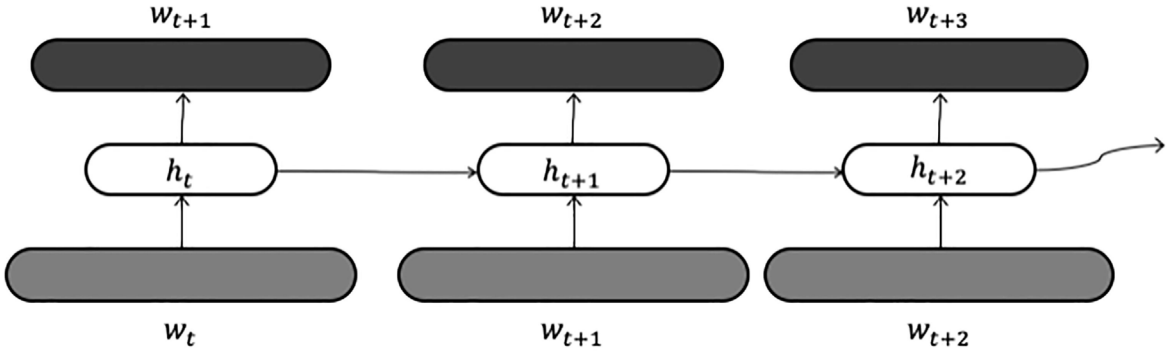

Fig. 3 (*continued*)

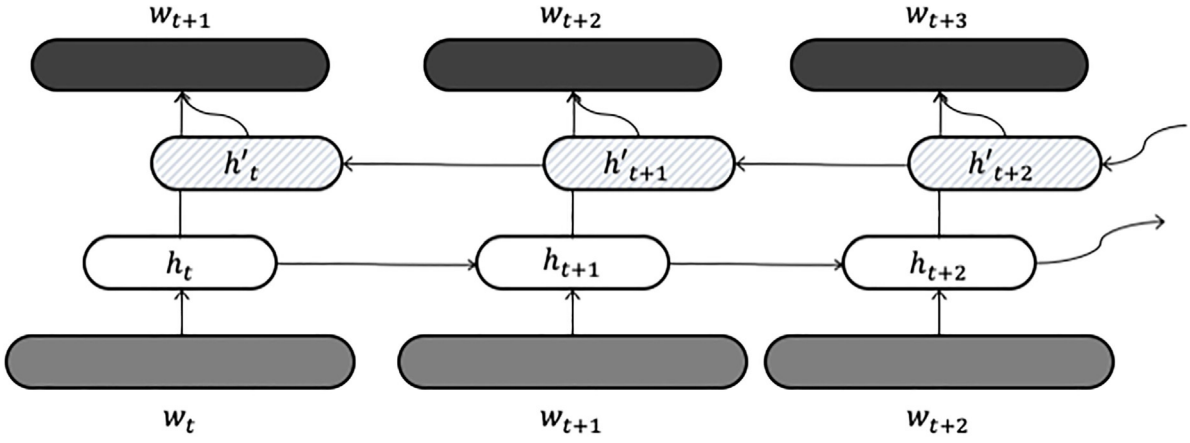

Fig. 3 (*continued*)

The authors compared word embedding techniques, clustered medical terms, clustered medical relations, and predicted semantic classes of medical terms (e.g., 'disease', 'symptom', 'treatment'). Comparing word embeddings and elementary vector representations (EVRs) showed that word embeddings produced better clusters of medical terms based on two metrics for cluster quality – the Davies-Bouldin Index (DBI), which measures intra-cluster distance vs. inter-cluster distance (higher values indicating better clusters) and the Dunn Index (DI), which estimates the optimum number of clusters based on their diameters and dissimilarity with other clusters. To analyze the results of medical term clustering, they compared the three types of word embeddings and found that the PubMed embeddings (followed by MedHelp) proved to provide the best representations for professional-oriented terms from UMLS, while for the consumer-oriented terms from Ranker, the performance of both the PubMed and MedHelp word embeddings are significantly better than using Wikipedia word embeddings.

To confirm that the word embeddings preserve medical relationships such as ⟨disease, treatment⟩ or ⟨disease, symptom⟩, the authors compared the word embeddings of the pairs, obtained from Ranker, to the word embeddings of the difference of the individual term, i.e. $V\langle a,b\rangle \triangleq V\langle b\rangle - V\langle a\rangle$.

Both PubMed and Medhelp embeddings show a clear separation of disease symptoms and treatment terms, while the Wikipedia embeddings fail to separate these classes. The authors also evaluated the embeddings for medical class prediction by calculating the centroids of the semantic clusters, and then predicting the semantic class of a query term based on the nearest centroid. This experiment also showed that the PubMed and MedHelp embeddings produce better

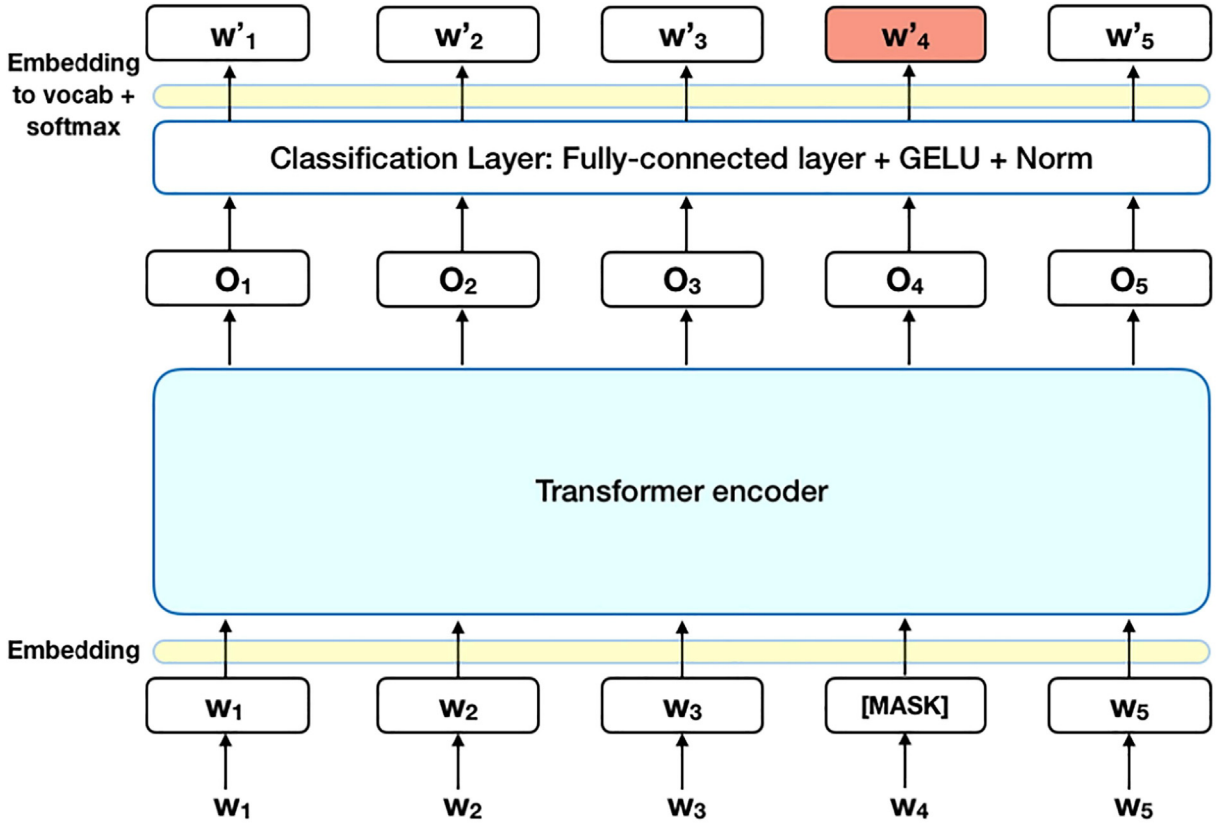

Fig. 4. BERT model (https://lyrn.ai/2018/11/07/explained-bert-state-of-the-art-language-model-for-nlp/). Taking masked input and outputting the masked word(s).

Table 2
Top ten similar words for common terms based on PMC word embeddings.

Word	Similar words
Doctor	Physician, dentist, receptionist, chiropractor, midwife, pharmacist, pediatrician, clinician, practitioner, friend
Pain	Discomfort, pains, fatigue, headache, backache, paresthesia, painful, dysesthesia, complaints, sensation
Surgery	Surgeries, surgical, elective, revisional, surgery., oesophagectomy, operation, TURP, esophagectomy, reoperative
Injury	Contusion, insult, ischemia, ischemic/reperfusion-induced, ischemia–reperfusion, injuries, ischemia/reperfusion, injury., traumatic, SCI
Medication	Medications, prescription, medication(s), antihypertensive, antihypertensives, prescribed, co-medication, prescriptions, prescribing, analgesics

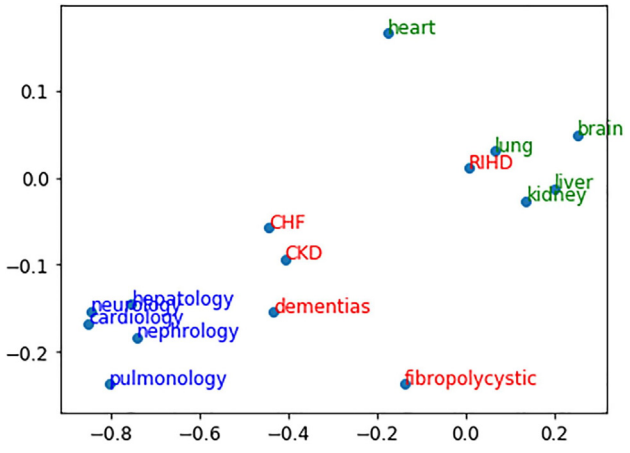

Fig. 5. Pubmed Word embedding visualization. Green text: Anatomical location (heart, lung, liver, kidney, brain). Red text: Diseases (RIHD i.e., Radiation-Induced Heart Disease, CHF i.e., Congestive Heart Failure, CKD i.e., Chronic kidney disease, dementias, fibropolycystic). Blue text: Medical specialties (cardiology, pulmonology, hepatology, nephrol.ogy, neurology).

results on the medical class prediction task (with accuracy scores at least 10% higher than with the Wikipedia embeddings), and that the results from the MedHelp word embeddings are comparable to the PubMed word embeddings.

De Vine et al. [30] used word embeddings for clinical information extraction. They used baseline features including orthographic, lexical, morphological, contextual, and external resource features (such as terms from SNOMED CT). They compared these to 'unsupervised' features including word embedding features (i.e., word2vec) and sequence features, which are constructed by concatenating normalized word vectors with normalized lexical vectors (vectors of *n*-grams and skip-grams). Each concatenated vector is normalized once again to obtain the sequence vector. De Vine et al. used K-means++ [77] to cluster the training data, and vectors in the test data were projected to the nearest cluster and assigned a cluster ID to be used as feature value. The baseline and unsupervised features are tested with conditional random fields (CRFs) on two clinical corpora: (i) concatenation of i2b2 train set [47], MedTrack [48], CLEF 2013 train and test sets, and (ii) i2b2 train set only and two non-clinical corpora – PubMed (from 2012) and Wikipedia (from 2009) [78]). Results based on word embedding features are comparable to baseline features. Also, adding word embedding features to the baseline features produced statistically significant improvements ($p < 0.05$) for F_1; however, the

Table 3
Types of corpora used to generate embeddings. Biomedical refers to either of: i) PubMed Central (PMC) articles or ii) PubMed articles. The news corpus is a pre-trained embedding provided by Google. Combined embeddings refers to any corpora generated by mixing of any of the above options. For combined corpora, we also listed the work in the individual corpora used in the combination.

Corpora	Task(s)
Clinical notes	Clinical information extraction [29–31,16], disease prediction [32–34], medical coding [35], patient phenotyping [36], re-admission prediction [37–39,17]
Biomedical	Clinical information extraction [29,30,40], medical coding [35]
News	Clinical information extraction [29]
Wikipedia	Clinical information extraction [29,30], medical coding [35]
Combined embeddings	Clinical information extraction [29], medical coding [35]

Table 4
Example of pre-processing. The input text has been lowercased, tokenized, lemmatized, and stripped of stop words and punctuation.

Example	
Text	*'Patient is suffering from high fever.'*
Pre-processed text	*'patient', 'suffer', 'high', 'fever'*

sequence features only showed a small improvement in the results. Through empirical evaluation, the authors also find that having *more* data is more effective than having *better formatted* data in the clinical domain. Moreover, the content, format, and domain of the data to derive the unsupervised features should be similar to the target corpus to get good results, and bio-medical data can be used instead of clinical data, for extracting the features without significant loss of information.

5.1.3. Intrinsic evaluation of clinical concept embeddings

For intrinsic evaluation of clinical concept embeddings, human similarity judgements are often used, such as MayoSRS [49] or UMNSRS [56,58]. In addition, Zhao et al.[43] found that embeddings trained on PubMed and DrugBank showed much higher correlation with human judgments in the UMNSRS datasets. They also performed coherence assessment and outlier detection, which showed that drug names were semantically close to other drug entities.

Yu et al.compared their results to human similarity judgments on MeSH terms from Nguyen et al. [79]. They also introduced a metric to determine the degree of semantic similarity between pairs of concepts. For evaluation, they calculated the similarity scores between 25 pairs of MeSH terms from the original word embeddings and the retrofitted embeddings, and found that the retrofitted embeddings trained on the UMLS-Similarity tool achieved the highest correlation with physician similarity judgments in terms of Spearman's rank correlation coefficient.

Choi et al.computed medical similarity and relatedness of nearest neighbors, and found that the concept embeddings from clinical notes (MCECN) best preserved the neighbourhood structure in terms of medical relatedness, and embeddings from OHSUMED performed the best in terms of medical similarity.

Huang et al.[17] tested their ClinicalBERT model on masked language modeling and observed a 30% improvement, and on next sentence prediction with a 19% improvement in accuracy over the original BERT results. They also evaluated that representation using a clinical concept dataset [51] (consisting of clinical term pairs with relatedness scores from physicians) and report Pearson correlation between (1) physician ratings of the clinical concept similarity from [51], and (2) cosine similarities between model embeddings (i.e., fastText ($r = 0.487$), word2vec ($r = 0.553$) and ClinicalBERT ($r = 0.670$)), where word2vec and fastText are trained on MIMIC-III for a fair comparison.

A qualitative analysis of the embeddings from the Alsentzer et al. [16] ClinicalBERT model also showed that it has greater cohesion around medical or clinic-operations terminology than BioBert.

5.2. Extrinsic evaluation – applications

In the previous section, we presented some intrinsic tests that can be used to evaluate the quality of created word embeddings, often in terms of correlations with human ratings of word similarity. However, as extensively documented, performance on intrinsic tests does not always correlate with performance on intended applications (i.e., so-called "extrinsic tasks").

Chiu et al. [66] investigated relationships between intrinsic and extrinsic evaluations of word embeddings on ten benchmark datasets by ranking word pairs based on the cosine similarity of the two words. They then compared the rankings to human rankings by calculating Spearman's rank correlation. For extrinsic evaluation, they used three standard sequence labeling tasks – part-of-speech tagging, chunking, and named entity recognition. The results demonstrated little to no correlation between intrinsic and extrinsic evaluation. While good scores on intrinsic evaluation tasks may demonstrate that the embeddings are capturing coherent information, it may not be useful for downstream tasks. So, while intrinsic evaluation can provide a good 'sanity check' of the embedding model, the performance on the final task is ultimately more important, and embedding models should be tuned accordingly.

In the clinical setting, extrinsic evaluation of word embeddings involves particular applications or tasks for which were are created. For example, if the goal is to predict bone fractures using doctor notes, analyzing the accuracy of such a system would be an extrinsic task. While specific tasks exist, there is no common data set that can be applied generally. Therefore, in this section, we cover various applications for which word embeddings have been used in order to clarify how word embeddings can be used for task automation in the clinical setting.

Once the word embeddings have been created (Section 4), and have been evaluated for semantic content (Section 5.1.1), they can be used to represent text documents, such as clinical notes, and input to machine learning classifiers for a variety of tasks. Often, each document can be represented as a matrix, where each row corresponds to the embedding for a word instance.

A common intermediate task is identifying clinical concepts in text. Si et al.[80] compared traditional word embeddings (word2vec, fastText, and GloVe) trained on MIMIC-III against ELMo, BERT, and BioBERT for clinical concept extraction on the i2b2 2010 and 2012 datasets (clincal notes with annotated concepts), and clinical reports with disease concepts from SemEval 2014 and 2015. The best results (which became the new state-of-the-art) were obtained by starting with the BERT-Large model, continuing training on MIMIC-III, and then adding the fine-tuning layer (a bi-LSTM) to the model. This model achieved F_1 scores between 0.80 and 0.90 on the 4 datasets, outperforming all other embedding methods. However, there is an available ELMo model trained on PubMed data which was not tested.

They also found that concept extraction performance decreased after a certain number of BERT fine-tuning iterations (about 340 k steps), potentially due to the model losing information learned from the general domain corpus and overfitting to the in-domain corpus. ELMo remained stable after about 280 k steps.

A common end task for word embeddings generated from clinical EMRs is predicting unplanned readmission after discharge [37–39]. Craig et al. [37] and Nguyen et al. [38] both used convolutional neural networks to make these predictions from clinical notes, while Pham et al. [39] used a dynamic memory model (specifically, an RNN). Although all three papers discussed different data, and report different performance metrics (from a 0.8 F_1 ccore to a 0.7 c-statistic), it is clear that these methods are useful and can be further developed for clinical settings for flagging follow-ups or more thorough testing before discharge.

Another common task is ICD code prediction. Patel et al. [35] used word embeddings trained on non-clinical notes (i.e., PubMed, PMC, Wikipedia, and combinations thereof) and observed an improvement of 1% in F_1 score on the automated coding review task by adapting the embeddings to the clinical space, while Escudié et al. [33] trained embeddings directly on the free-text of EMRs.

Table 5
Available embeddings for clinical data and concepts. Since ELMo models use character information and BERT models use sub-word information, they can generate a representation for any concept.

Name	Model	Data/Concepts	Terms	Dim.
PubMed-w2v.bin[a]	word2vec	PubMed	2.4 M	200
PMC-w2v.bin[b]	word2vec	PubMed Central	2.5 M	200
PubMed-and-PMC-w2v.bin[c]	word2vec	PubMed, PubMed Central	4.1 M	200
wikipedia-pubmed-and-PMC-w2v.bin[d]	word2vec	PubMed, PubMed Central, Wikipedia	5.5 M	200
drug word embeddings[e]	word2vec	PubMed, DrugBank	553,195	420
AWE-CM [49]	word2vec	UMLS CUI (concepts)	265 M	300
claims_codes_hs_300 [55]	word2vec	ICD-9 codes (concepts)	51,327	300
claims_cuis_hs_300 [55]	word2vec	UMLS CUI (concepts)	14,852	300
cui2vec [56]	word2vec/GloVe	UMLS CUI (concepts)	108,477	500
concept embeddings [58]	AiTextML	MeSH ID (concepts)	26,103	100
word embeddings [58]	AiTextML	PubMed	513,196	100
ELMo (PubMed model) [11]	ELMo	PubMed	NA	1024
BioBERT [15]	BERT	PubMed	NA	768/1024
ClinicalBERT [16,17]	BERT	MIMIC III	NA	768

[a] http://evexdb.org/pmresources/ngrams/PubMed/.
[b] http://evexdb.org/pmresources/ngrams/PMC/.
[c] http://evexdb.org/pmresources/vec-space-models/wikipedia-pubmed-and-PMC-w2v.bin.
[d] http://evexdb.org/pmresources/vec-space-models/wikipedia-pubmed-and-PMC-w2v.bin.
[e] https://github.com/chop-dbhi/drug_word_embeddings.

Table 6
Available datasets for intrinsic evaluation.

Name	Metric	Data	Pairs
SimLex-999	similarity	word pairs (free association [71])	999
WordSim353	relatedness	word pairs (general)	353
MEN	relatedness	word pairs (Wikipedia + web)	3,000
UMNSRS-Similarity	similarity	UMLS concept pairs	566
UMNSRS-Relatedness	relatedness	UMLS concept pairs	588
MayoSRS	relatedness	SnoMedCT concept pairs	101

Additionally, in a similar vein to ICD code prediction, Gehrmann et al. [36] performed patient phenotyping (i.e., predicting 10 conditions from advanced cancer to depression) with F_1 scores in the mid-80s. Such a system can help streamline the care provided by doctors by suggesting ICD codes in decision support systems.

Word embeddings have been applied to other clinical predictive tasks – many on publicly available datasets, which enables reproducible comparisons between systems. Wang et al. [29] studied the effect of different corpora on the tasks of hand, finger, and wrist fracture detection and of predicting the smoking status of a patient using i2b2's (Informatics for Integrating Biology to the Bedside) 2006 dataset [47]. Using a different i2b2 dataset (i.e., the 2008 "Obesity Challenge"), Dubois et al. [32] predicted different patient phenotypes ranging from asthma to obesity. Kholgi et al. [34] and De Vine et al. [30] used the 2010 i2b2 dataset for clinical information extraction.

Word embeddings can also be used for information retrieval tasks such as named entity recognition (NER). After training an ELMo model, Zhu et al. [61] performed NER using a RNN followed by a conditional random field (CRF), which was trained on i2b2 dataset.

They compared the model with state-of-the-art models on the i2b2 dataset and found an improvement of 3.4% in F_1 score.

Zhao et al. [43] trained three versions of a bi-directional RNN for drug name recognition and classification, using the following types of word embeddings as input: (1) fixed, pre-trained embeddings treated as constants, (2) variable word embeddings initialized with pre-trained embeddings but treated as learnable parameters updated by the model, and (3) randomly initialized word embeddings which are then updated by the model. The results showed that all types of RNN perform well, but the versions that considered word embeddings as learnable parameters produced better results. In the case of drug name classification, the RNN with pre-trained learnable word embeddings showed slightly better performance as compared to the fixed word embeddings.

Huang et al.[17] tested their ClinicalBERT model on downstream tasks such as predicting hospital readmission within 30 days using the MIMIC-III dataset. After pre-processing, the final cohort consisted of approximately 34 K patients. Since ClinicalBERT uses fixed-length input, the notes were divided into sub-sequences. They used both discharge summaries and early clinical notes for hospital readmission prediction. ClinicalBERT outperformed the state-of-the-art in both cases. They also showed that ClinicalBERT provides interpretable predictions, by revealing which terms in clinical notes were predictive of patient readmission based on the model's self-attention mechanism.

Alsentzer et al.[16] evaluated their ClinicalBERT model on two named entity recognition (NER) tasks (concept extraction and entity extraction), two de-identification tasks, and a natural language inference task (NLI); they showed that their model performed better than the original BERT and BioBERT on NER and NLI tasks, but worse on de-identification. They posited that this is because MIMIC is already de-identified, with personal health information (PHI) replaced with placeholder tokens, which affects training and representation.

Beltagy et al.[19] evaluated SciBERT on NER, participant intervention comparison outcome extraction (PICO), classification, relation classification, and dependency parsing. The empirical results showed that SciBERT often significantly outperformed BERT-Base with minimal task-specific architectures and without fine-tuning.

As demonstrated in the aforementioned set of applications, word embeddings are a useful and versatile tool with the ability to perform well in many predictive tasks. Their utility extends to (often noisy) clinical note data. Word embeddings can capture valuable information contained in clinical free-text, at relatively low cost (without the need for manual annotation or expert curation).

6. Limitations of word embeddings

Although word embeddings are widely used for generating vector representations of clinical text data, they still have limitations.

6.1. Evaluating word embeddings

Evaluations can often be inconclusive, since the performance of word embeddings can vary greatly given different tasks [72]. Therefore, it is important to compare word embeddings in terms of both intrinsic *and* extrinsic measures (see Table 7).

Furthermore, evaluation can be inconsistent since different datasets are used, making it difficult to compare different clinical word embeddings. To solve this problem, Peng et al.[40] created the Biomedical Language Understanding Evaluation (BLUE), a benchmark consisting of five tasks (i.e., sentence similarity, named entity recognition, relation extraction, document multi-label classification and inference). The tasks are derived from ten existing datasets ranging in genre

Table 7
Examples of word embeddings trained on clinical data. We report the source *corpora* (e.g., clinical notes, biomedical articles) on which the word embeddings are trained on, as well as the *model* used. We also include the *intrinsic* and *extrinsic* tasks on which these embeddings were evaluated.

	Word embeddings		Evaluation	
Paper	Corpora	Model	Intrinsic	Extrinsic
De Vine et al., 2015 [30]		word2vec		clinical information extraction
Chiu et al., 2016 [72]	PMC, Pubmed, PMC + Pubmed	word2vec	relatedness, similarity	NER
Dubois et al., 2017 [32]	text notes, OHSUMED	GloVe	–	disease prediction, mortality prediction
Wang et al., 2018 [29]	Mayo Clinic text notes, PubMed, Wikipedia, Google News	GloVe	similarity (qualitative),	clinical information extraction, reaction extraction
Huang et al., 2016 [46]	MedHelp online forum, PubMed, Wikipedia		cluster quality evaluation	
Choi et al., 2016 [55]	OHSUMED, medical claims	word2vec	conceptual similarity, medical relatedness	–
Yu et al., 2016 [52])	UMLS and MeSH terms	LDA	UMLS-Similarity	–
Mencia et al., 2016 [58]	BioASQ, PubMed	All-in-text	MiniMayoSRS, UMNSRS similarity/relatedness	–
Boag et al., 2017 [49]	MIMIC-III	word2vec	similarity	–
Patel et al., 2017 [35]	PubMed, medical claims	word2vec	medical term similarity	medical coding review
Beam et al., 2018 [56]	PubMed, medical claims, UMLS semantic types	word2vec, Glove	comorbidity, causative, and drug-conditions relations and UMNSRS similarity/relatedness	
Zhao et al., 2018 [43]	PubMed, DrugBank	word2vec	UMNSRS similarity/relatedness	drug name recognition/classification
Craig et al., 2017 [37]	discharge notes	word2vec (Skipgram)	—	30-day unplanned prediction
Nguyen et al., 2017 [38]	hospital patient records	random init, word2vec	cluster evaluation	unplanned readmission within 6 months prediction
Pham et al., 2016 [39]	hospital patient records	random init + advanced techniques	—	unplanned readmission prediction, high-risk patient prediction
Escudié et al., 2018 [33]	electronic health records from hospital	three-layer stack of denoising autoencoders	—	disease prediction
Gehrmann et al., 2018 [36]	discharge summaries from MIMIC-III	word2vec	word similarity	phenotype classification
Wang et al., 2018 [29]	clinical notes from hospital, PMC, News, Wikipedia + Gigaword	word2vec (Skip-gram)	word/semantic similarity	information extraction, smoking status prediction, fracture detection,
Kholgi et al., 2016 [34]	i2b2/VA 2010 [81], ShARe/CLEF 2013 eHealth Evaluation Lab [82]	word2vec (Skip-gram)	—	disease prediction, term extraction

Table 8
Examples of word embeddings trained on clinical data (continued).

	Word embeddings		Evaluation	
Paper	Corpora	Model	Intrinsic	Extrinsic
Devlin et al., 2018 [12]	BookCorpus + Wikipedia	BERT		
Peters et al., 2018 [11]	PubMed	ELMo		
Alsentzer et al., 2019 [16]	MIMIC-III (all notes, discharge summaries only)	BERT	—	NER (concept extraction, entity extraction), de-identification, NLI
Beltagy et al., 2019 [19]	papers from Semantic Scholar	BERT	—	NER, PICO, classification, relation classification, dependency parsing
Huang et al., 2019 [17]	MIMIC-III	BERT	relatedness between clinical concepts	30 day hospital readmission
Lee et al., 2019 [15]	PubMed, PMC	BERT	—	NER, question answering, relation extraction
Peng et al., 2019 [40]	PubMed, MIMIC-III	BERT, ELMo	—	sentence similarity, NER, relation extraction, document multilabel classification, NLI

(i.e., biomedical abstracts and articles, clinical text notes), size, and difficulty. The aim of BLUE is to provide a standardized benchmark to allow for comparison of different models. A model's performance on the BLUE benchmark is determined by a macro-average of their F_1 scores and Pearson correlation scores on the ten tasks.

6.2. Bias

Since word embeddings are trained on real-world data, they may learn whatever social or cultural biases exist in those data [83]. Bolukbasi et al. [84] identified gender bias in the publicly available GloVe word embeddings by geometrically evaluating the word embeddings and checking for analogies that were considered to be stereotypical by annotators. For example, with word embeddings trained on the Google News dataset, they reported that certain occupations have a tendency to be associated with a specific genders and descriptors (e.g., "he" to "doctor", "brilliant", and "shopkeeper", and "she" to "nurse", "lovely", and "housewife"). The authors proposed a debiasing algorithm consisting of two steps: 1) identifying the direction of the word embedding containing 'gender definitional' information (i.e., words that *must* contain aspects of gender, such as "he" and "nephew"), and 2) removing bias by pushing all other words to the origin according to the aforementioned dimension of gender.

Word embeddings generally perpetuate biases present in real-world data. Previous work [85] showed real-world biases in the treatment of patients in clinical settings, with African American patients being systematically under-treated for pain relative to white Americans. How these real-world biases manifest in embeddings, and how to correct them without negatively impacting results is also an active area of research.

6.3. Interpretability

Although the internals of many machine learning models are treated opaquely, it is often important to provide the reasoning for any clinical judgments that a model makes. Indeed, it may even be *required* under certain interpretations of jurisdictional law [86]. However, providing a rationale for a model's decisions based on word embeddings is not straightforward.

Senel et al. [87] investigated the distribution of semantic categories in word embeddings in order to measure interpretability of the individual dimensions. Here, interpretability is measured with a *word intrusion test* where, for each word embedding dimension, human annotators are presented with a set of the five top-ranked words (based on the value of the target dimension for each word) plus one bottom-ranked word (i.e., the intruder word). Human annotators are then asked to identify the intruder word in a random ordering of the set. The authors also measured the *encoding level* (i.e., 'concept level') in each word embedding dimension by comparing a distribution from which the concept words are sampled, and another from which all other words are sampled. These comparisons were done using the Bhattacharyya distance [88], which measures the similarity of two probability distributions. While their categorical decompositions of word embedding dimensions showed correlations with semantic categories, this evaluation requires human-defined clusters of semantic concepts with labels.

Another issue with interpretability is that most word embedding models encode (and do not distinguish) notions of both similarity and relatedness (e.g., *fever* and *flu* are related but are not similar in meaning) whereas, in a medical context, we may favour one notion over the other. For example, we may be interested in all related symptoms of a disease instead of similar words or synonyms. Although some evaluation sets, such as UMNSRS, have separate measures for relatedness and similarity judgments, distinguishing between similar and related words in these models remains an open research challenge.

6.4. Privacy issues for clinical data

Privacy has been a major concern in healthcare, but it is especially important for clinical data. Culnane et al. [89] experimented on medical records and billing information (not converted to word embeddings), and were able to identify a participant based on publicly available information including Wikipedia and news articles. They were able to identify famous women based on their dates of birth and number of children along with their respective dates of birth. The identification was even easier in the women who gave birth late in life due to the small cohort size. Similarly, famous athletes were identified based on publicly available dates of birth and major injuries information. The authors also experimented with the effects of perturbing the dates by a few weeks to preserve privacy but it did not show any significant improvement.

Weggenmann et al. reported privacy issues in using vectors of word-document information. They showed that removing the personal information of the author of a document does not necessarily result in confidentiality, and that term-frequency vectors can be used to re-identify the author in special settings.

More research is needed as to how well word embeddings can preserve privacy for clinical data. This question is of high importance because it is very helpful for the research community to use pre-trained clinical word embeddings, but such embeddings should not be shared before privacy is ensured. *Differential privacy* [90] is an area of research dealing with maximizing accuracy while minimizing privacy violations, and may form the basis for such activity.

6.5. Lack of publicly available clinical word embeddings

Currently, there are only a few publicly available pre-trained word embeddings models for general contexts. One of the main issues for clinical data is that data privacy policies often prevent the release of any models learned from those data. The result is that researchers are forced to create their own models using their own data, and it is difficult to compare results when each research group has a different dataset and different word embedding models. This may lead to a crisis of non-reproducibility in the community. More research on the potential impacts of these word embedding models, including bias, interpretability, and privacy, would allow researchers to share models more openly.

7. Conclusion

In this paper, we provide a guide (Fig. 2) for training and using word embeddings for machine learning tasks in the clinical domain, as well as an overview of existing work on word embeddings for clinical text. Algorithm 1 summarizes the procedure for training a word embeddings.

Algorithm 1. Training and using word embeddings on clinical text.

1: **Choose training data** from existing sources (Table 3 and Section 3.1) or use your own data.
2: **Pre-process** the data (Section 3.2).
3: **Choose the algorithm** for training word embeddings (Table 1 and Section 2).
4: **Train** word embeddings or BERT/ELMo model (Section 4).
5: Perform intrinsic **evaluation** if desired (Table 6 and Section 5.1.1).
6: **if** using BERT/ELMo
7: **Apply** the trained model to the target data.
8: **else**
9: **Transform** target data to word embeddings using the trained word embeddings (Section 4.4).
10: **Apply** the target task model to the embeddings.
11: Perform extrinsic **evaluation** on the target task (Section 5).

When training word embeddings for clinical text, it is important to consider several factors, including: the size of the training corpus, the domain of the training corpus, the characteristics of the word embedding model, and the quality of the trained embeddings. For quick bootstrapping, there are several pre-trained embeddings that can often be easily applied to desired target data (see Table 5).

While word embeddings have enabled advances in natural language processing for healthcare, more research is needed to fully understand what the models represent, as well as the potential biases and privacy issues therein. However, for classification and prediction tasks in the clinical domain, using unsupervised text representations such as word embeddings can be faster to implement and provide better results than traditional feature engineering.

Declaration of Competing Interest

The authors declared that there is no conflict of interest.

Acknowledgements

Rudzicz is a CIFAR Chair in Artificial Intelligence.

Appendix A. Types of word embeddings

Traditional representations map words to sparse, high dimensional vectors with a dimensionality of V, i.e., one element per distinct word type in the vocabulary. In contrast, prediction-based word embedding methods (e.g., word2vec, FastText, Poincaré, and ELMo) map each word in a vocabulary to dense representations whose dimensionality is typically much lower than the size of the vocabulary. These approaches are enumerated below.

(a) **One-hot encoding**
A one-hot encoding is a binary vector representation treating the text as categorical data. In this model, each word is represented by a vector of length equal to the total number of unique words in the corpus, where each position in the vector corresponds to a unique word in the vocabulary. The values in this vector are all 0, except for the position corresponding to the desired word, which is set to 1. A document can also be represented as a single vector where all words that appear in the document are set to 1.
A Python implementation is available from scikit-learn[27].

(b) **Term Frequency (TF)**
Also known as a count vector, a term frequency vector represents each document as a single vector containing the frequencies of the words that appear in the document. All linguistic information conveyed in a given document is represented by a single vector, where each dimension denotes the frequency of occurence of the corresponding word in the document.

(c) **Term Frequency-Inverse Document Frequency (TF-IDF)**
The TF-IDF model is an extension of the TF model where the observed counts are weighted by inverse document frequency (IDF). Application of the IDF weights acts to smooth observed frequency counts, ultimately reducing the effect of words that are common across many documents. In the TF model, only the counts of the words are used, which results in the model giving more weight to more common but less discriminatory words such as “the” or “an”. The TF-IDF vector for a term t is document d is as follows:

$$\mathrm{TF-IDF}(t,d) = TF(t,d) * IDF(t),$$

where

$$TF = \frac{\text{frequency of } t \text{ in } d}{\text{total number of terms in } d}$$

$$IDF = log\left(\frac{\mathrm{N}}{\mathrm{n}}\right)$$

where N is the total number of documents and n is the number of documents a term t has appeared in. The terms are usually n-grams. An n-gram is a sequence of n number of adjacent words (or characters) in a given text. A Python implementation available in scikit-learn[28].

(e) **Word2vec** Word2vec is word embedding toolkit that implements two different embedding methods: the Continuous Bag of Words (CBOW) model and the Skip-gram (SG) model [3–5]. Both the CBOW and SG models are examples of neural embedding models. That is, both models employ shallow neural network architectures to learn the parameters of the embedding vectors. Although the two models have similar architectures and approaches to parameter learning (i.e. stochastic gradient descent), the specific loss functions are unique, reflecting

[27] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html.
[28] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html.

the distinct objectives of the respective models. Detailed derivations of the parameter update equations for both CBOW and SG models can be found in [6]. Relationships between word2vec embeddings (e.g. embeddings learned from "prediction" models) and GloVe embeddings (e.g. embeddings learned from "count" models) are discussed in [91,92].

Continuous Bag of Words (CBOW)

In the CBOW model, the objective function involves predicting a focal word given its context. The CBOW model is essentially a log-linear classification model with multinomial/softmax loss function. The goal is to determine parameters of the embedding vectors that are most probable under the following model:

$$P(w_f|w_c) = \frac{exp\left(w_f^T w_c\right)}{\sum_{i=1}^{V} exp(w_i^T w_c)}$$

where w_f is the focal word, w_c is the context (one or more words), and V is the size of the vocabulary.

The simplest case involves a single word context. At which point the hidden representation/layer is merely the vector representation of the context word. The inner product between the context word vectors and the focal word vectors can be seen as assigning a score. High scores map to high predicted probabilities under the model. The goal is to assign high scores to focal words that are likely under the context.

The CBOW model can be extended to multi-word contexts, at which point the hidden representation is chosen to be the element-wise sum/average of the specific context word vectors. Given this hidden representation, the inner product between the latent context representation and the focal word vector can again be seen as assigning a score. The goal is to assign high scores to focal words that are likely under the multi-word context.

Skip-gram Model

The Skip-gram model is complementary to the CBOW model in the sense that its objective function involves predicting a context word(s) given a single focal word.

$$P(w_c|w_f) = \sum_{c=1}^{C} \frac{exp(w_c^T w_f)}{\sum_{i=1}^{V} exp(w_c^T w_i)}$$

In the case of a single word context the model is identical to CBOW; however, for multi-word contexts the objective becomes (slightly) more complicated. As there exists only a single word context, again the hidden layer simply copies the current vector representation of the focal word. However, the objective involves now predicting C context words. As such, the model loss/objective function is a composition (sum) of the respective context-specific loss functions. Here there will be C inner-product scores; one for each focal-word context-word pair. The goal will be to assign high scores to context words that are likely under the given focal word.

CBOW and SG share many commonalities in their model architectures and the way they are trained. At the time of writing, distributed vector representations of words are common in the machine learning and natural language processing communities. However, most successful applications are learned using deep neural network architectures that are slow to train on consumer hardware. The chief contributions of the series of Mikolov papers [3–5] are largely algorithmic (i.e. demonstrating that useful vector embeddings could be learned from shallow neural networks architectures in a computationally efficient manner). Specific contributions involve the incorporation of a hierarchical softmax loss function that is based on Huffman encodings, as well as methods to (negatively) sub-sample frequently occurring pairs of focal/context words. The original papers also provide an evaluation of the sensitivity of the models to the values of important hyper-parameters. Two key parameters for this model include: the size of the embedding dimension (as is true with all word vector methods), and also the size/implementation of context window (how many words, is it applied symmetrically about the focal word, etc.). Additional hyper-parameters of interest include: the negative sampling rate and the neural network optimization parameters (SGD variant chosen, learning rate, etc.). Both the CBOW and skip-gram models are implemented by Gensim[29] and the Google code archive[30].

Variants of word2vec

Doc2vec and paragraph2vec [7] are variants of the word2vec that generate vectors for documents or paragraphs instead of words. Word2vec is trained to predict the surrounding words while doc2vec/paragraph2vec focuses on predicting words in the document/paragraph. Unlike word2vec that predicts the surrounding words only based on the given word, doc2vec also relies on the document to make the prediction of the words in the document. Doc2vec has two types: Paragraph Vector-Distributed Memory (PV-DM) model that is similar to skip-gram and Paragraph Vector-Distributed Bag of Words (PV-DBOW) that is similar to CBOW model.

(f) **Global Vectors (GloVe)**

A co-occurrence matrix is an V by V matrix where V is the vocabulary size (n.b. the vocabulary is the set of all n-grams in the text-corpus). Each entry of the matrix corresponds to the number of times any two n-grams from the vocabulary co-occur, within a pre-specified discrete context window, measured across the entire corpus.

The GloVe model was proposed by Pennington et al. [8] as a means for learning word vector embeddings. In contrast to the word2vec approaches discussed above, GloVe uses global information from the term co-occurrence matrix (TCM) to learn word embeddings. The goal of the GloVe model is to learn vector embeddings such as to minimize the reconstruction error between co-occurrence statistics predicted by the model and global co-occurrence statistics observed in the training corpus. The specific loss function which is minimized is:

$$L = \sum_{i=1}^{N}\sum_{j=1}^{N} f(X_{ij})\left(w_i^T w_j + b_i + b_j - X_{ij}\right)^2$$

X_{ij} represents the observed co-occurrence count from the empirical term co-occurrence matrix. w_i and w_j represent the vector embeddings of word i and word j, respectively, and b_i and b_j represent offsets or biases for word i and word j, respectively. And $f(\cdot)$ is a function defined by the authors as:

[29] https://radimrehurek.com/gensim.

[30] https://code.google.com/archive/p/word2vec/.

$$f(\cdot) = \begin{cases} \left(\frac{x}{x_{max}}\right)^{\alpha} & \text{if } x > x_{max} \\ 1 & \text{otherwise} \end{cases}$$

The model is trained using stochastic gradient descent (or some variant). Again, the model consists of numerous hyper-parameters that must be judiciously chosen, including: vector embedding dimension, context window size, value of x_{max} (the authors suggest $x_{max} = 100$), value of alpha (the authors suggest 3/4), and various optimization specific hyper-parameters (choice of SGD variant, learning rate, etc.). Code can be found here[31].

(g) **FastText**

FastText builds on a specific limitation of the word2vec, GloVe and other popular vector embedding models, namely, their inability to handle out-of-vocabulary (OOV) terms (i.e. any n-gram that is not present in system's vocabulary). FastText extends the word2vec Skip-gram (SG) model by considering internal sub-word information [9]. In particular, FastText builds a vector representation for a given word as the composition of its morphological components. Practically, this can be accomplished by viewing a word as a composition of character level n-grams. Working at a character level allows the FastText model to share morphological information across words, and also to construct vector representations of words that are rare or perhaps never seen at all in the training corpus. In the event that character level n-grams are unable to construct a given word, the model typically assigns a 0-vector to the word under consideration.

The objective is essentially the same as the SG model discussed above, where the goal is to predict word contexts given a focal word. The distinction for the FastText model being that both the vectors for focal words and the context word(s) are represented as the composition of their character level n-grams. The objective function is:

$$\sum_{t=1}^{T}\left[\sum_{c\in C_t} l(s(w_t, w_c)) + \sum_{n\in N_{t,c}} l(-s(w_t, n))\right]$$

where t is the position in the text and l is the logistic loss function defined as

$$l(x) = log(1 + e^{-x})$$

and s is the scoring function that computes the similarity between a word w and a context c:

$$s(w, c) = \sum_{g\in G_w} z_g^T v_c$$

where G_w is the set of character n-grams in the word w and z_g is the vector representation of n-gram g, and v_c is the context vector. The first term in the objective function considers the context words as positive examples, and the second term samples negative examples randomly from the dictionary.

As the FastText model is very similar to the word2vec SG model, many of the comments about hyper-paramter tuning made above apply directly to this context. An additional hyper-parameter that must be specified in the FastText model is the character n-gram size. In their original paper, Bojanowski et al. [9] suggest that n-gram size of 3–6 provides sufficient sub-word information for most evaluations they performed. The code is available in C++[32].

(h) **Poincaré Embedding**

Most word embeddings are represented in the Euclidean space. Word vectors in Euclidean space are sometimes unable to capture hierarchical structure observed in certain corpus (or may require high dimensional embedding dimensions in order to capture this complexity). Poincaré embeddings [21] represent the data in the hyperbolic space (or more precisely Poincaré ball). Unlike Euclidean space, hyperbolic space has a constant negative curvature, which helps to capture the hierarchy and similarity of the data more accurately. The hyperbolic distance between the points u & v in the Poincaré ball is defined as,

$$d(u, v) = arcosh\left(1 + 2\frac{||u - v||^2}{\left(1 - ||u||^2\right)\left(1 - ||v||^2\right)}\right)$$

where *arcosh* is inverse cosine hyperbolic function and ||.|| denoted the Euclidean distance metric. If u and v are away from the boundary of the ball, d(u,v) is smaller and if they are closer to the boundary it is larger thus preserving the hierarchical/tree-like structure between points. The Poincare embedding optimization function is defined as,

$$\Theta' \leftarrow argmin_{\Theta}\ \mathscr{L}(\Theta) \quad s.t.\ \forall\theta_i \in \Theta : ||\theta_i|| < 1$$

where Θ denotes the word embedding. The loss function $\mathscr{L}(\Theta)$ is defined as,

$$\mathscr{L}(\Theta) = \sum_{(u,v)\in\mathscr{D}} log\frac{e^{-d(u,v)}}{\sum_{v'\in\mathscr{N}(u)} e^{-d(u,v')}}$$

where $\mathscr{D} = (u, v)$ denotes a set of hypernymy relations and $\mathscr{N}(u)$ is the set negative examples for u.

The main advantage of this embedding is that less number of dimensions can capture the hierarchical information. A Python implementation of Poincaré embeddings is available from Gensim[33].

(i) **Embeddings from Language Models (ELMo)**

ELMo [11] is a contextual character-level embedding method. Most word embeddings predict a focal word given its context or a (set of) context words given a focal word. In contast, ELMo word representations are computed using a bi-directional RNN language model with

[31] https://nlp.stanford.edu/projects/glove/.
[32] https://fasttext.cc/.
[33] https://radimrehurek.com/gensim/models/poincare.html.

character convolutions over the entire sentence. The weighted sum of the original word vector and all the layer representation (both forward and backward) are estimated. ELMo vectors can be used by concatenating them to any word embeddings such as word2vec and Fastext. Code and embeddings pretrained on a large corpus are freely available[34].

(j) **Probabilistic FastText**

Probabilistic FastText[10] combines Gaussian mixture model and FastText. Each word is represented as a Gaussian mixture model, defined as

$$f(x) = \sum_{i=1}^{K} p_{w,i}\, \mathcal{N}\left(x; \vec{\mu}_{w,i}, \sum_{w,i}\right)$$

where K is the number of components, $\mathcal{N}$ denotes the Gaussian distribution, $\{\mu_{w,i}\}_{k=1}^{K}$ are the mean vectors, $\left\{\sum_{w,i}\right\}$ are the covariance matrices and $\{p_{w,i}\}_{k=1}^{K}$ are the component probabilities. The K components represent K different senses of a word.

The mean is estimated using the subword structure using n-grams of characters in the word w,

$$\mu_w = \frac{1}{|NG_w| + 1}\left(v_w + \sum_{g \in NG_w} z_g\right)$$

where v_w dictionary word representation of word w, $|NG_w|$ is the number of n-grams of the word w and z_g is a vector associated with the n-gram g. To train the model they maximize the similarity between the words in the context window and minimize the similarity with a random word. The loss function is defined as

$$L(w, c, c') = max[0, m - log\, E(w, c) + log\, E(w, c')]$$

where E denotes a similarity function, c is the context word and c' denotes the random word. This function learns the parameters dictionary vectors $\{v_w\}_{i=1}^{K}$ and character n-gram vectors z_g by pushing the log of similarity of context pair (w, c) higher than a negative context pair (w, c') by margin m.

FastText is able to capture the sub-word structure, different word senses and provides better representation of rare/unseen words, however this method has not been used in the biomedical domain. The code is available[35].

(k) **Bidirectional Encoder Representations for Transformers (BERT)**

BERT [12] consists of a multi-layer bidirectional transformer encoder. It differentiates itself from other conditional language models through *bidirectional* pre-training. During pre-training, BERT is trained on the unsupervised 'masked language modeling and 'next sentence prediction' tasks. BERT can be easily fine-tuned for various tasks by adding a task-specific output layer.

Tokenization: BERT uses the WordPiece tokenizer [93], which tokenizes based on the following steps:

1. Initialize the word unit vocabulary with all the individual characters in the language.
2. Build a language model on the training data using the vocabulary from 1.
3. Update the vocabulary with the most frequent combinations of the existing words in the vocabulary iteratively, until a predefined limit of word units is reached or the likelihood increase falls below a certain threshold.

Pre-training: BERT is trained using two unsupervised tasks: binarized next sentence prediction and masked language modeling. Conditional language models are trained left-to-right or right-to-left, since training them in a bidirectional manner would allow each word to "see itself". To circumvent this problem, while at the same training the language model bidirectionally, BERT uses *masked* language modeling. 15% of the tokens are randomly masked (i.e., replaced with the "[MASK]" token), and the model is trained to predict the masked tokens.

Pre-trained Models: Two model sizes for BERT are available:

1. BERT-Base with 12 layers of transformer blocks, 12 attention heads, and 110 million parameters.
2. BERT-Large with 24 layers of transformer blocks, 16 attention heads and, 340 million parameters.

Appendix B. Medical relatedness

The Medical Conceptual Similarity Measure (MCSM) is defined as:

$$MCSM(V, T, k) = \frac{1}{V(T)} \sum_{v \in V(T)} \sum_{i=1}^{k} \frac{1_T(v(i))}{log_2(i+1)}$$

, where $V(T)$ is the set of concepts of type T, $v(i)$ is the ith closest neighbor of v, k is the size of the neighborhood, and "1_T is an indicator function which is 1 if concept v(i) is of type T, and 0 otherwise". MRM is defined as:

$$MRM(V, T, k) = \frac{1}{|V*|} \sum_{v \in V*} 1_R\left(\bigcup_{i=1}^{k}(v - s)(i)\right)$$

, where "1_R is the indicator function which returns 1 if any of the medical concepts in the top-k neighborhood of the selected medical concept is an element with the given relation R, and 0 otherwise" [55].

[34] https://allennlp.org/elmo.

[35] https://github.com/benathi/multisense-prob-fasttext.

References

[1] R. Leaman, R. Khare, Z. Lu, Challenges in clinical natural language processing for automated disorder normalization, J. Biomed. Inform. 57 (2015) 28–37, https://doi.org/10.1016/j.jbi.2015.07.010. <http://www.sciencedirect.com/science/article/pii/S1532046415001501> .

[2] S. McDonald, M. Ramscar, Testing the distributioanl hypothesis: The influence of context on judgements of semantic similarity, Proceedings of the Annual Meeting of the Cognitive Science Society, vol. 23, 2001.

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

[4] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Adv. Neural Informat. Process. Syst., 2013, pp. 3111–3119.

[5] T. Mikolov, W.-T. Yih, G. Zweig, Linguistic regularities in continuous space word representations, in: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2013, pp. 746–751.

[6] X. Rong, word2vec parameter learning explained, arXiv preprint arXiv:1411.2738.

[7] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International Conference on Machine Learning, 2014, pp. 1188–1196.

[8] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.

[9] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, Enriching word vectors with subword information, arXiv preprint arXiv:1607.04606.

[10] B. Athiwaratkun, A.G. Wilson, A. Anandkumar, Probabilistic fasttext for multi-sense word embeddings, arXiv preprint arXiv:1806.02901.

[11] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: Proc. of NAACL, 2018.

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

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Lukasz Kaiser, I. Polosukhin, Attention is all you need, in: Adv. Neural Informat. Process. Syst., 2017, pp. 5998–6008.

[14] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, S. Fidler, Aligning books and movies: Towards story-like visual explanations by watching movies and reading books, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 19–27.

[15] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C.H. So, J. Kang, Biobert: pre-trained biomedical language representation model for biomedical text mining, arXiv preprint arXiv:1901.08746.

[16] E. Alsentzer, J. Murphy, W. Boag, W.-H. Weng, D. Jindi, T. Naumann, M. McDermott, Publicly available clinical BERT embeddings, in: Proceedings of the 2nd Clinical Natural Language Processing Workshop, Association for Computational Linguistics, Minneapolis, Minnesota, USA, 2019, pp. 72–78, https://doi.org/10.18653/v1/W19-1909. <https://www.aclweb.org/anthology/W19-1909> .

[17] K. Huang, J. Altosaar, R. Ranganath, Clinicalbert: Modeling clinical notes and predicting hospital readmission, arXiv preprint arXiv:1904.05342.

[18] A.E.W. Johnson, T.J. Pollard, L. Shen, L.-W. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, R.G. Mark, L.A. Celi, R.G. Mark, MIMIC-III, a freely accessible critical care database, Sci. Data 3 (2016) 160035, https://doi.org/10.1038/sdata.2016.35. http://10.0.4.14/sdata.2016.35 .

[19] I. Beltagy, A. Cohan, K. Lo, Scibert: Pretrained contextualized embeddings for scientific text, arXiv preprint arXiv:1903.10676.

[20] W. Ammar, D. Groeneveld, C. Bhagavatula, I. Beltagy, M. Crawford, D. Downey, J. Dunkelberger, A. Elgohary, S. Feldman, V. Ha, et al., Construction of the literature graph in semantic scholar, arXiv preprint arXiv:1805.02262.

[21] M. Nickel, D. Kiela, Poincaré embeddings for learning hierarchical representations, in: Adv. Neural Informat. Process. Syst., 2017, pp. 6338–6347.

[22] J. Howard, S. Ruder, Universal language model fine-tuning for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, 2018, pp. 328–339.

[23] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving language understanding with unsupervised learning, Tech. Rep., Technical Report, OpenAI, 2018.

[24] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, OpenAI Blog 1 (8) (2019).

[25] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q.V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, arXiv preprint arXiv:1906.08237.

[26] G. Lample, A. Conneau, Cross-lingual language model pretraining, arXiv preprint arXiv:1901.07291.

[27] Y. Sun, S. Wang, Y. Li, S. Feng, X. Chen, H. Zhang, X. Tian, D. Zhu, H. Tian, H. Wu, Ernie: Enhanced representation through knowledge integration, arXiv preprint arXiv:1904.09223.

[28] L.v.d. Maaten, G. Hinton, Visualizing data using t-sne, J. Machine Learn. Res. 9 (Nov) (2008) 2579–2605.

[29] Y. Wang, S. Liu, N. Afzal, M. Rastegar-Mojarad, L. Wang, F. Shen, H. Liu, A comparison of word embeddings for the biomedical natural language processing, arXiv preprint arXiv:1802.00400.

[30] L. De Vine, M. Kholghi, G. Zuccon, L. Sitbon, A. Nguyen, Analysis of word embeddings and sequence features for clinical information extraction, 2015.

[31] H.-C. Shin, L. Lu, L. Kim, A. Seff, J. Yao, R.M. Summers, Interleaved text/image deep mining on a very large-scale radiology database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1090–1099.

[32] S. Dubois, N. Romano, Learning effective embeddings from medical notes, arXiv preprint arXiv:1705.07025.

[33] J.-B. Escudié, A. Saade, A. Coucke, M. Lelarge, Deep representation for patient visits from electronic health records, arXiv preprint arXiv:1803.09533.

[34] M. Kholghi, L. De Vine, L. Sitbon, G. Zuccon, A. Nguyen, The benefits of word embeddings features for active learning in clinical information extraction, in: Proceedings of the Australasian Language Technology Association Workshop 2016, 2016, pp. 25–34.

[35] K. Patel, D. Patel, M. Golakiya, P. Bhattacharyya, N. Birari, Adapting pre-trained word embeddings for use in medical coding, BioNLP 2017 (2017) 302–306.

[36] S. Gehrmann, F. Dernoncourt, Y. Li, E.T. Carlson, J.T. Wu, J. Welt, J. Foote Jr., E.T. Moseley, D.W. Grant, P.D. Tyler, et al, Comparing deep learning and concept extraction based methods for patient phenotyping from clinical narratives, PloS One 13 (2) (2018) e0192360.

[37] E. Craig, C. Arias, D. Gillman, Predicting readmission risk from doctors' notes, arXiv preprint arXiv:1711.10663.

[38] P. Nguyen, T. Tran, N. Wickramasinghe, S. Venkatesh, Deepr: A convolutional net for medical records, IEEE J. Biomed. Health Informat. 21 (1) (2017) 22–30.

[39] T. Pham, T. Tran, D. Phung, S. Venkatesh, Deepcare: A deep dynamic memory model for predictive medicine, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2016, pp. 30–41.

[40] Y. Peng, S. Yan, Z. Lu, Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets, arXiv preprint arXiv:1906.05474.

[41] S. Moen, T.S.S. Ananiadou, Distributional semantics resources for biomedical text processing, in: Proceedings of the 5th International Symposium on Languages in Biology and Medicine, Tokyo, Japan, 2013, pp. 39–43.

[42] Y. Zhu, E. Yan, F. Wang, Semantic relatedness and similarity of biomedical terms: examining the effects of recency, size, and section of biomedical publications on the performance of word2vec, BMC Med. Inform. Decis. Mak. 17 (1) (2017) 95, https://doi.org/10.1186/s12911-017-0498-1. <http://www.ncbi.nlm.nih.gov/pubmed/28673289> . <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5496182> .

[43] M. Zhao, A.J. Masino, C.C. Yang, A framework for developing and evaluating word embeddings of drug-named entity, in: Proceedings of the BioNLP 2018, 2018, pp. 156–160.

[44] G. Szarvas, V. Vincze, R. Farkas, J. Csirik, The bioscope corpus: annotation for negation, uncertainty and their scope in biomedical texts, in: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing, Association for Computational Linguistics, 2008, pp. 38–45.

[45] W.W. Chapman, W. Bridewell, P. Hanbury, G.F. Cooper, B.G. Buchanan, A simple algorithm for identifying negated findings and diseases in discharge summaries, J. Biomed. Informat. 34 (5) (2001) 301–310.

[46] J. Huang, K. Xu, V.V. Vydiswaran, Analyzing multiple medical corpora using word embedding, in: 2016 IEEE International Conference on Healthcare Informatics (ICHI), IEEE, 2016, pp. 527–533.

[47] Ö. Uzuner, I. Goldstein, Y. Luo, I. Kohane, Identifying patient smoking status from medical discharge records, J. Am. Med. Inform. Assoc. 15 (1) (2008) 14–24.

[48] E.M. Voorhees, W.R. Hersh, Overview of the trec 2012 medical records track, in: TREC, 2012.

[49] W. Boag, H. Kané, AWE-CM Vectors: Augmenting Word Embeddings with a Clinical Metathesaurus arXiv:1712.01460. http://arxiv.org/abs/1712.01460.

[50] O. Levy, Y. Goldberg, Dependency-based word embeddings, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2014, pp. 302–308. https://doi.org/10.3115/v1/P14-2050. http://aclweb.org/anthology/P14-2050.

[51] T. Pedersen, S.V. Pakhomov, S. Patwardhan, C.G. Chute, Measures of semantic similarity and relatedness in the biomedical domain, J. Biomed. Informat. 40 (3) (2007) 288–299.

[52] Z. Yu, T. Cohen, E.V. Bernstam, T.R. Johnson, B.C. Wallace, Retrofitting Word Vectors of MeSH Terms to Improve Semantic Similarity Measures (2016) 43–51. <https://aclweb.org/anthology/W/W16/W16-6106.pdf> .

[53] M. Faruqui, J. Dodge, S.K. Jauhar, C. Dyer, E. Hovy, N.A. Smith, Retrofitting word vectors to semantic lexicons, in: Proceedings of NAACL-HLT, 2015.

[54] B.T. McInnes, T. Pedersen, S.V.S. Pakhomov, UMLS-Interface and UMLS-Similarity: open source software for measuring paths and semantic similarity, vol. 2009, American Medical Informatics Association, 2009, pp. 431–435. http://www.ncbi.nlm.nih.gov/pubmed/20351894, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2815481.

[55] Y. Choi, C.Y.-I. Chiu, D. Sontag, Learning Low-Dimensional Representations of Medical Concepts, vol. 2016, American Medical Informatics Association, 2016, pp. 41. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5001761/.

[56] A.L. Beam, B. Kompa, I. Fried, N. Palmer, X. Shi, T. Cai, I.S. Kohane, Clinical Concept Embeddings Learned from Massive Sources of Medical Data, arXiv, 2018, pp. 1–27 arXiv:1804.01486. URL http://arxiv.org/abs/1804.01486.

[57] S.G. Finlayson, P. LePendu, N.H. Shah, Building the graph of medicine from millions of clinical narratives, Sci. Data 1 (140032) (2014) 1–9, https://doi.org/10.1038/sdata.2014.32. <http://www.nature.com/articles/sdata201432> .

[58] E.L. Mencia, G. de Melo, J. Nam, Medical Concept Embeddings via Labeled Background Corpora, 2016, pp. 4629–4636. URL http://www.lrec-conf.org/proceedings/lrec2016/pdf/1190_Paper.pdf.

[59] J. Nam, E.L. Mencía, J. Fürnkranz, All-in Text: learning document, label, and word representations jointly, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016, pp. 1948–1954.
[60] S. Pakhomov, B. McInnes, T. Adams, Y. Liu, T. Pedersen, G. Melton, Semantic similarity and relatedness between clinical terms: An experimental study, in: Proceedings of the Annual Symposium of the American Medical Informatics Association, 2010.
[61] H. Zhu, I.C. Paschalidis, A. Tahmasebi, Clinical concept extraction with contextual word embedding, arXiv preprint arXiv:1810.10566.
[62] J. Rogers, O. Bodenreider, Snomed ct: Browsing the browsers, in: KR-MED, 2008, pp. 30–36.
[63] F. Hill, R. Reichart, A. Korhonen, Simlex-999: Evaluating semantic models with (Genuine) similarity estimation, Comput. Linguist. 41 (4) (2015) 665–695, https://doi.org/10.1162/COLI_a_00237. <http://www.mitpressjournals.org/doi/10.1162/COLI_a_00237> .
[64] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Pasca, A. Soroa, A study on similarity and relatedness using distributional and wordnet-based approaches, in: Proceedings of NAACL-HLT 2009, (2009).
[65] E. Bruni, N.K. Tran, B. Marco, Multimodal distributional semantics, J. Artif. Intell. Res. 49 (December) (2013) 1–47.
[66] B. Chiu, A. Korhonen, S. Pyysalo, Intrinsic evaluation of word vectors fails to predict extrinsic performance, in: Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP, 2016, pp. 1–6.
[67] Y. Tsvetkov, M. Faruqui, W. Ling, G. Lample, C. Dyer, Evaluation of Word Vector Representations by Subspace Alignment, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17-21 September 2015 (September) (2015), 2015, pp. 2049–2054, https://doi.org/10.18653/v1/D15-1243.
[68] G.A. Miller, C. Leacock, R. Tengi, R.T. Bunker, A semantic concordance, in: Proceedings of Human Language Technologies, 1993, pp. 303–308.
[69] C. Fellbaum, WordNet: An Electronic Lexical Database, MIT Press, Cambridge, MA, 1998.
[70] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2013), 2013.
[71] D. Nelson, C. McEvoy, T. Schreiber, The university of south florida word association, rhyme, and word fragment norms. http://www.usf.edu/FreeAssociation/.
[72] B. Chiu, G. Crichton, A. Korhonen, S. Pyysalo, How to train good word embeddings for biomedical nlp, in: Proceedings of the 15th Workshop on Biomedical Natural Language Processing, 2016, pp. 166–174.
[73] L. Smith, L.K. Tanabe, R.J. nee Ando, C.-J. Kuo, I.-F. Chung, C.-N. Hsu, Y.-S. Lin, R. Klinger, C.M. Friedrich, K. Ganchev, et al, Overview of biocreative ii gene mention recognition, Genome Biol. 9 (2) (2008) S2.
[74] J.-D. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, N. Collier, Introduction to the bio-entity recognition task at jnlpba, in: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, Association for Computational Linguistics, 2004, pp. 70–75.
[75] A. Hliaoutakis, Semantic similarity measures in mesh ontology and their application to information retrieval on medline, Master's thesis, 2005.
[76] S.V. Pakhomov, T. Pedersen, B. McInnes, G.B. Melton, A. Ruggieri, C.G. Chute, Towards a framework for developing semantic relatedness reference standards, J. Biomed. Informat. 44 (2) (2011) 251–265.
[77] D. Arthur, S. Vassilvitskii, k-means+ +: The advantages of careful seeding, in: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.
[78] C.M. De Vries, R. Nayak, S. Kutty, S. Geva, A. Tagarelli, Overview of the inex 2010 xml mining track: Clustering and classification of xml documents, in: International Workshop of the Initiative for the Evaluation of XML Retrieval, Springer, 2010, pp. 363–376.
[79] H. Nguyen, H. Al-Mubaid, New ontology-based semantic similarity measure for the biomedical domain, 2006, pp. 623 – 628. https://doi.org/10.1109/GRC.2006.1635880.
[80] Y. Si, J. Wang, H. Xu, K. Roberts, Enhancing Clinical Concept Extraction with Contextual Embedding, JAMIA (in press) arXiv:1902.08691. http://arxiv.org/abs/1902.08691.
[81] Ö. Uzuner, B.R. South, S. Shen, S.L. DuVall, 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text, J. Am. Med. Inform. Assoc. 18 (5) (2011) 552–556.
[82] S. Pradhan, N. Elhadad, B.R. South, D. Martinez, L.M. Christensen, A. Vogel, H. Suominen, W.W. Chapman, G.K. Savova, Task 1: Share/clef ehealth evaluation lab 2013, in: CLEF (Working Notes), 2013.
[83] A.C. Kozlowski, M. Taddy, J.A. Evans, The geometry of culture: Analyzing meaning through word embeddings, arXiv preprint arXiv:1803.09288.
[84] T. Bolukbasi, K.-W. Chang, J.Y. Zou, V. Saligrama, A.T. Kalai, Man is to computer programmer as woman is to homemaker? debiasing word embeddings, in: Advances in Neural Information Processing Systems, 2016, pp. 4349–4357.
[85] K.M. Hoffman, S. Trawalter, J.R. Axt, M.N. Oliver, Racial bias in pain assessment and treatment recommendations, and false beliefs about biological differences between blacks and whites, Proc. Nat. Acad. Sci. 113 (16) (2016) 4296–4301.
[86] F. Doshi-Velez, M. Kortz, R. Budish, C. Bavitz, S.J. Gershman, D. O'Brien, S. Shieber, J. Waldo, D. Weinberger, A. Wood, Accountability of AI Under the Law: The Role of Explanation, 2017. arXiv:1711.01134, doi:10.2139/ssrn.3064761.
[87] L.K. Şenel, İhsan Utlu, V. Yücesoy, A. Koç, T. Çukur, Semantic structure and interpretability of word embeddings, IEEE/ACM Trans. Audio Speech Language Process. (2018).
[88] A. Bhattacharyya, On a measure of divergence between two statistical populations defined by their probability distributions, Bull. Calcutta Math. Soc. 35 (1943) 99–109.
[89] C. Culnane, B.I.P. Rubinstein, V. Teague, Health data in an open world, CoRR abs/1712.05627. arXiv:1712.05627. http://arxiv.org/abs/1712.05627.
[90] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: Theory Cryptography Conference, Springer, 2006, pp. 265–284.
[91] M. Baroni, G. Dinu, G. Kruszewski, Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, 2014, pp. 238–247.
[92] O. Levy, Y. Goldberg, Neural word embedding as implicit matrix factorization, in: Adv. Neural Informat. Process. Syst., 2014, pp. 2177–2185.
[93] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: Bridging the gap between human and machine translation, arXiv preprint arXiv:1609.08144.