Embeddings for the Lexicon: Modelling and Representation

Christian Chiarcos[1] **Thierry Declerck**[2] **Maxim Ionov**[1]
[1] Applied Computational Linguistics Lab, Goethe University, Frankfurt am Main, Germany
[2] DFKI GmbH, Multilinguality and Language Technology, Saarbrücken, Germany
{chiarcos,ionov}@informatik.uni-frankfurt.de
declerck@dfki.de

Abstract

In this paper, we propose an RDF model for representing embeddings for elements in lexical knowledge graphs (e.g. words and concepts), harmonizing two major paradigms in computational semantics on the level of representation, i.e., distributional semantics (using numerical vector representations), and lexical semantics (employing lexical knowledge graphs). Our model is a part of a larger effort to extend a community standard for representing lexical resources, OntoLex-Lemon, with the means to connect it to corpus data. By allowing a conjoint modelling of embeddings and lexical knowledge graphs as its part, we hope to contribute to the further consolidation of the field, its methodologies and the accessibility of the respective resources.

1 Background

With the rise of word and concept embeddings, lexical units at all levels of granularity have been subject to various approaches to embed them into numerical feature spaces, giving rise to a myriad of datasets with pre-trained embeddings generated with different algorithms that can be freely used in various NLP applications. With this paper, we present the current state of an effort to connect these embeddings with lexical knowledge graphs.

This effort is a part of an extension of a widely used community standard for representing, linking and publishing lexical resources on the web, OntoLex-Lemon[1]. Our work aims to complement the emerging OntoLex module for representing Frequency, Attestation and Corpus Information (FrAC) which is currently being developed by the W3C Community Group "Ontology-Lexica", as presented in [4]. There we addressed only frequency and attestations, whereas core aspects of corpus-based information such as embeddings were identified as a topic for future developments. Here, we describe possible use-cases for the latter and present our current model for this.

2 Sharing Embeddings on the Web

Although word embeddings are often calculated on the fly, the community recognizes the importance of pre-trained embeddings as these are readily available (it saves time), and cover large quantities of text (their replication would be energy- and time-intense). Finally, a benefit of re-using embeddings is that they can be grounded in a well-defined, and, possibly, shared feature space, whereas locally built embeddings (whose creation involves an element of randomness) would reside in an isolate feature space. This is particularly important in the context of multilingual applications, where collections of embeddings are in a single feature space (e.g., in MUSE [6]).

Sharing embeddings, especially if calculated over a large amount of data, is not only an economical and ecological requirement, but unavoidable in many cases. For these, we suggest to apply established web standards to provide metadata about the lexical component of such data. We are not aware of any provider of pre-trained word embeddings which come with machine-readable metadata. One such example is language information: while ISO-639 codes are sometimes being used for this purpose, they are not given in a machine-readable way, but rather documented in human-readable form or given implicitly as part of file names.[2]

2.1 Concept Embeddings

It is to be noted, however, that our focus is not so much on word embeddings, since lexical information in this context is apparently trivial – plain

[1] https://www.w3.org/2016/05/ontolex/

[2] See the ISO-639-1 code 'en' in FastText/MUSE files such as https://dl.fbaipublicfiles.com/arrival/vectors/wiki.multi.en.vec.

tokens without any lexical information do not seem to require a structured approach to lexical semantics. This changes drastically for embeddings of more abstract lexical entities, e.g., word senses or concepts [17], that need to be synchronized between the embedding store and the lexical knowledge graph by which they are defined. WordNet [14] synset identifiers are a notorious example for the instability of concepts between different versions: Synset `00019837-a` means 'incapable of being put up with' in WordNet 1.71, but 'established by authority' in version 2.1. In WordNet 3.0, the first synset has the ID `02435671-a`, the second `00179035-s`.[3]

The precompiled synset embeddings provided with AutoExtend [17] illustrate the consequences: The synset IDs *seem* to refer to WordNet 2.1 (`wn-2.1-00001742-n`), but use an ad hoc notation and are not in a machine-readable format. More importantly, however, there *is* no synset `00001742-n` in Princeton WordNet 2.1. This *does* exist in Princeton WordNet 1.71, and in fact, it seems that the authors actually used this edition instead of the version 2.1 they refer to in their paper. Machine-readable metadata would not prevent such mistakes, but it would facilitate their verifiability. Given the current conventions in the field, such mistakes will very likely go unnoticed, thus leading to highly unexpected results in applications developed on this basis. Our suggestion here is to use resolvable URIs as concept identifiers, and if they provide machine-readable lexical data, lexical information about concept embeddings can be more easily verified and (this is another application) integrated with predictions from distributional semantics. Indeed, the WordNet community has adopted OntoLex-Lemon as an RDF-based representation schema and developed the Collaborative Interlingual Index (ILI or CILI) [3] to establish sense mappings across a large number of WordNets. Reference to ILI URIs would allow to retrieve the lexical information behind a particular concept embedding, as the WordNet can be queried for the lexemes this concept (synset) is associated with. A versioning mismatch can then be easily spotted by comparing the cosine distance between the word embeddings of these lexemes and the embedding of the concept presumably derived from them.

[3] See https://github.com/globalwordnet/ili

2.2 Organizing Contextualized Embeddings

A related challenge is the organization of contextualized embeddings that are not adequately identified by a string (say, a word form), but only by that string in a particular context. By providing a reference vocabulary to organize contextualized embeddings together with the respective context, this challenge will be overcome, as well.

As a concrete use case of such information, consider a classical approach to Word Sense Disambiguation: The Lesk algorithm [12] uses a bag-of-words model to assess the similarity of a word in a given context (to be classified) with example sentences or definitions provided *for different senses* of that word in a lexical resource (training data, provided, for example, by resources such as WordNet, thesauri, or conventional dictionaries). The approach was very influential, although it always suffered from data sparsity issues, as it relied on string matches between a very limited collection of reference words and context words. With distributional semantics and word embeddings, such sparsity issues are easily overcome as literal matches are no longer required.

Indeed, more recent methods such as BERT allow us to induce contextualized word embeddings [20]. So, given only a single example for a particular word sense, this would be a basis for a more informed word sense disambiguation. With more than one example per word sense, this is becoming a little bit more difficult, as these now need to be either aggregated into a single sense vector, or linked to the particular word sense they pertain to (which will require a set of vectors as a data structure rather than a vector). For data of the second type, we suggest to follow existing models for representing attestations (glosses, examples) for lexical entities, and to represent contextualized embeddings (together with their context) as properties of attestations.

3 Addressing the Modelling Challenge

In order to meet these requirements, we propose a formalism that allows the conjoint publication of lexical knowledge graphs and embedding stores. We assume that future approaches to access and publish embeddings on the web will be largely in line with high-level methods that abstract from details of the actual format and provide a conceptual view on the data set as a whole, as provided, for ex-

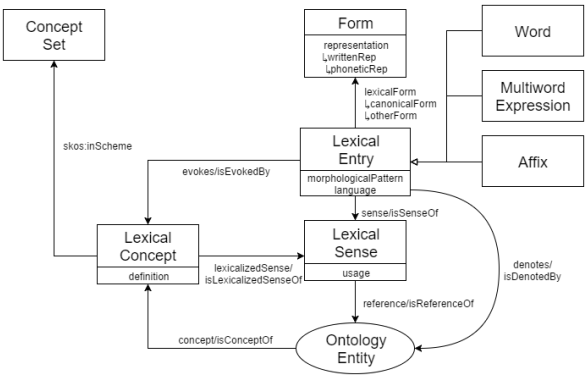

Figure 1: OntoLex-Lemon core model

ample, by the `torchtext` package of PyTorch.[4]

There is no need to limit ourselves to the currently dominating tabular structure of commonly used formats to exchange embeddings; access to (and the complexity of) the actual data will be hidden from the user.

A promising framework for the conjoint publication of embeddings and lexical knowledge graph is, for example, RDF-HDT [9], a compact binary serialization of RDF graphs and the literal values these comprise. We would assume it to be integrated in programming workflows in a way similar to program-specific binary formats such as Numpy arrays in pickle [8].

3.1 Modelling Lexical Resources

Formalisms to represent lexical resources are manifold and have been a topic of discussion within the language resource community for decades, with standards such as LMF [10], or TEI-Dict [19] designed for the electronic edition and/or search in individual dictionaries. Lexical data, however, does not exist in isolation, and synergies can be unleashed if information from different dictionaries is combined, e.g., for bootstrapping new bilingual dictionaries for languages X and Z by using another language Y and existing dictionaries for $X \mapsto Y$ and $Y \mapsto Z$ as a pivot.

Information integration beyond the level of individual dictionaries has thus become an important concern in the language resource community. One way to achieve this is to represent this data as a knowledge graph. The primary community standard for publishing lexical knowledge graphs on the web is OntoLex-Lemon [5].

OntoLex-Lemon defines four main classes of lexical entities, i.e., concepts in a lexical resource: (1) **LexicalEntry** representation of a lexeme in a lexical knowledge graph, groups together form(s) and sense(s), resp. concept(s) of a particular expression; (2) (lexical) **Form**, written representation(s) of a particular lexical entry, with (optional) grammatical information; (3) **LexicalSense** word sense of one particular lexical entry; (4) **LexicalConcept** elements of meaning with different lexicalizations, e.g., WordNet synsets.

As the dominant vocabulary for lexical knowledge graphs on the web of data, the OntoLex-Lemon model has found wide adaptation beyond its original focus on ontology lexicalization. In the WordNet Collaborative Interlingual Index [3] mentioned before, OntoLex vocabulary is used to provide a single interlingual identifier for every concept in every WordNet language as well as machine-readable information about it (including links with various languages).

To broaden the spectrum of possible usages of the core model, various extensions have been developed by the community effort. This includes the emerging module for frequency, attestation and corpus-based information, FrAC.

The vocabulary elements introduced with FrAC cover frequency and attestations, with other aspects of corpus-based information described as prospective developments in our previous work [4]. Notable aspects of corpus information to be covered in such a module for the purpose of lexicographic applications include contextual similarity, co-occurrence information and embeddings.

Because of the enormous technical relevance of the latter in language technology and AI, and because of the requirements identified above for the publication of embedding information over the web, this paper focuses on embeddings,

3.2 OntoLex-FrAC

FrAC aims to (1) extend OntoLex with corpus information to address challenges in lexicography, (2) model lexical and distributional-semantic resources (dictionaries, embeddings) as RDF graphs, (3) provide an abstract model of relevant concepts in distributional semantics that facilitates applications that integrate both lexical and distributional information. Figure 2 illustrates the FrAC concepts and properties for frequency and attestations (gray) along with new additions (blue) for embeddings as a major form of corpus-based information.

Prior to the introduction of embeddings, the

[4] https://pytorch.org/text/

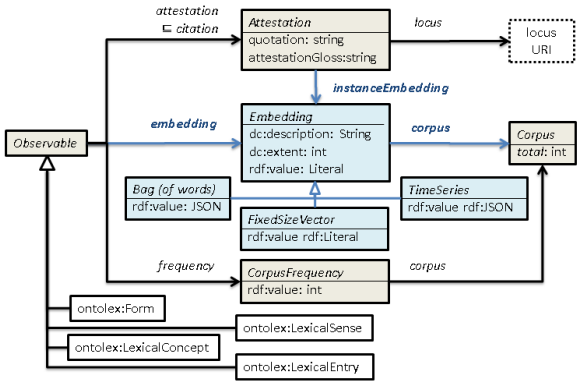

Figure 2: OntoLex-FrAC overview: Extensions for embeddings highlighted in blue

main classes of FrAC were (1) **Observable**, any unit within a lexical resource about which information can be found in a corpus, includes all main classes of OntoLex-Core, lexical forms, lexical entries, lexical senses and concepts; (2) **Corpus**, a collection of linguistic data from which empirical evidence can be derived, including corpora in the sense as understood in language technology; (3) **Attestation**, example for a specific phenomenon, usage or form found in a corpus or in the literature; (4) **CorpusFrequency**, absolute frequency of a particular observation in a given corpus.

4 Modelling Embeddings in FrAC

In the context of OntoLex, word embeddings are to be seen as a feature of individual lexical forms. However, in many cases, word embeddings are not calculated from plain strings, but from normalized strings, e.g., lemmatized text. For such data, we model every individual lemma as an `ontolex:LexicalEntry`. Moreover, as argued in Sec. 2, embeddings are equally relevant for lexical senses and lexical concepts; the `embedding` property that associates a lexical entity with an embedding is thus applicable to every `Observable`.

4.1 Word Embeddings

Pre-trained word embeddings are often distributed as text files consisting of the label (token) and a sequence of whitespace-separated numbers. E.g. the entry for the word *frak* from the GloVe embeddings [15]:

```
frak 0.015246 -0.30472 0.68107 ...
```

Since our focus on publishing and sharing embeddings, we propose to provide the value of an embedding as a literal `rdf:value`. If necessary, more elaborate representations, e.g., using `rdf:List`, may subsequently be generated from these literals.

A natural and effort-less modelling choice is to represent embedding values as string literals with whitespace-separated numbers. For decoding and verification, such a representation benefits from metadata about the length of the vector. For a fixed-size vector, this should be provided by `dc:extent`. An alternative is an encoding as JSON list. In order to support both structured and string literals, FrAC does not restrict the type of the `rdf:value` of embeddings.

Lexicalized embeddings should be published together with their metadata, at least procedure/method (`dct:description` with free text, e.g., *"CBOW"*, *"SKIP-GRAM"*, *"collocation counts"*), data basis (`frac:corpus`), and dimensionality (`dct:extent`).

We thus introduce the concept embedding, with a designated subclass for fixed-size vectors:

Embedding (Class) is a representation of a given `Observable` in a numerical feature space. It is defined by the methodology used for creating it (`dct:description`), the URI of the corpus or language resource from which it was created (`corpus`). The literal value of an Embedding is provided by `rdf:value`.

`Embedding` ⊑ `rdf:value exactly 1` ⊓ `corpus exactly 1` ⊓ `dct:description min 1`

embedding (ObjectProperty) is a relation that maps an `Observable` into a numerical feature space. An embedding is a structure-preserving mapping in the sense that it encodes and preserves contextual features of a particular `Observable` (or, an aggregation over all its attestations) in a particular corpus.

FixedSizeVector (Class) is an `Embedding` that represents a particular `Observable` as a list of numerical values in a k-dimensional feature space. The property `dc:extent` defines k.

For the GloVe example, a lemma (lexical entry) embedding can be represented as follows:

```
:frak a ontolex:LexicalEntry;
  ontolex:canonicalForm/
    ontolex:writtenRep "frak"@en;
  frac:embedding [
    a frac:FixedSizeVector;
      rdf:value "0.015246 ...";
```

```
    dct:source
      <https://catalog.ldc....>;
    dct:extent 50^^^xsd:int;
    dct:description "GloVe v.1.1,
        ..."@en. ].
```

4.2 Contextualized Embeddings

Above, we mentioned contextualized embeddings, and more recent methods such as ELMo [16], Flair NLP [1], or BERT [7] have been shown to be remarkably effective at many NLP problems.

In the context of lexical semantics, contextual embeddings can prove beneficial for inducing or distinguishing word senses, and in extension of the classical Lesk algorithm, for example, a lexical sense can be described by means of the contextualized word embeddings for the examples associated with that particular lexical sense, and words for which word sense disambiguation is to be performed can then just be compared with these. These examples then serve a similar function as attestations in a dictionary, and indeed, the link has been emphasized before [11]. Within FrAC, contextualized embeddings are thus naturally modelled as a property of `Attestation`.

instanceEmbedding (ObjectProperty) for a given `Attestation`. The property `instanceEmbedding` provides an embedding of the example in its current corpus context into a numerical feature space (see `Embedding`).

In this modelling, multiple contextualized embeddings can be associated with, say, a lexical sense by means of an attestation that then points to the actual string context. Considering *play*, multiple WordNet examples (glosses) per sense can thus be rendered by different fixed-size vectors:

```
wn31:play_n a ontolex:LexicalEntry;
  ontolex:sense wn31:07032045-n,
    wn31:play_n_4, ...
wn31:07032045-n
    a ontolex:LexicalSense;
    frac:attestation [
        frac:quotation "the play
        lasted two hours";
        frac:locus wn31:07032045-n;
        frac:instanceEmbedding
            wn31-bert:07032045-n-1
        ].
wn31-bert:07032045-n-1 a
    frac:FixedSizeVector;
    dc:extent "300"^^xsd:int;
    rdf:value "0.327246 0.48170 ...";
    dc:description "...";
    frac:corpus <http://wordnet-rdf.
       princeton.edu/static/wordnet.
       nt.gz> .
```

Like most RDF models, this appears to be overly verbose, but by introducing a subclass for a set of embeddings from which the embedding can inherit extent, corpus, and description information, e.g. `:WN31FixedSizeVector`, the BERT embedding in this example becomes much more digestable – without any loss in information:

```
wn31-bert:07032045-n-1 a
    :WN31FixedSizeVector;
    rdf:value "0.327246 0.48170 ...".
```

4.3 Other Types of Embeddings

FrAC is not restricted to uses in language technology; its definition of 'embedding' is thus broader than the conventional interpretation of the term in NLP, and based on its more general usage across multiple disciplines. In mathematics, the embedding f of an object X in another object Y ($f : X \to Y$) is defined as an injective, structure-preserving map (morphism). Important in the context of FrAC is the structure-preserving character of the projection into numerical feature spaces, as embeddings are most typically used to assess similarity, e.g., by means of cosine measure, the entire point being that these assessments remain stable when cooccurrence vectors are projected into a lower-dimensional space.

In computational lexicography, raw collocation counts continue to be used for the same purpose. In its classical form, these collocation counts get aggregated into bags of words, sometimes augmented with numerical weights. Bags of words are closely related to raw collocation vectors (and thus, prototypical embeddings), with the main difference being that collocation vectors consider only a finite set of diagnostic collocates (the reference dictionary), whereas a bag of words can include *any* word that occurs in the immediate context of the target expression.

In this understanding, a bag of words can be considered as an infinite-dimensional collocation vector, i.e., as an embedding in the broad sense introduced above. The practical motivation is that applications of bag-of-words models and fixed-size vectors are similar, and that bags of words remain practically important to a significant group of OntoLex or FrAC users.

A difference is that a bag of words model, if it provides frequency information or another form of numerical scores, must not be modelled by a plain list, but rather, by a dictionary or a map. As a data structure for this, we recommend JSON literals.

To represent this model, we introduce the following `Embedding` subclass:

BagOfWords (Class) is an `Embedding` that represents an `Observable` by a set of collocates or their mapping to numerical scores.

Another type of embeddings concerns sequential data, and one example for that are multimodal corpora. In a case study with Leiden University, we explored the encoding of dictionaries for African sign languages. In addition to graphics and videos, such dictionaries can also contain sensor data that records the position of hands and digits during the course of a sign or a full conversation. To search in this data, conventional techniques like dynamic time warp [2] are being applied – effectively in analogy with cosine distance among finite-size vectors. Furthermore, such data has also been addressed in NLP research in the context of neural time series transformation for the sake of translation between or from sign languages [18]. Also, in an NLP context, time series analysis is relevant for stream processing, so this would make this data structure of more general interest [13] in and beyond the language technology community.

Both from a modelling perspective and in terms of actual uses of such forms of lexical data, it is thus appealing to extend the concept of embeddings to time series data. In our current use case, we assume that time series data is mostly relevant in relation to attestations rather than OntoLex core concepts, but we can foresee that generalizations over multiple attestations could be developed at some point which would then used to extrapolate prototypical time series information for canonical forms, lexical entries, senses, or concepts.

To represent this, we introduce another `Embedding` subclass:

TimeSeries (Class) is an `Embedding` that represents an `Observable` or its `Attestation` as a sequence of a fixed number of data points recorded over a certain period of time.

The `dc:extent` of a time series specifies the number of data points per observation. The `rdf:value` should be a structured JSON literal.

5 Summary and Outlook

We identified a number of shortcomings in the (lack of) standards applied by providers of pretrained embeddings on the web and aim to address them by providing a vocabulary that covers the relevant aspects of lexical data involved in this context, grounded in existing specifications for lexical metadata and publication forms for lexical knowledge graphs on the web of data. We provide an RDF vocabulary for encoding numerical, corpus-based information, in particular, embeddings such as those prevalent in language technology, in conjunction with, or as a part of lexical knowledge graphs. We cover a broad range of applications from NLP to computational lexicography, and a broad range of data structures that fall under a generalized understanding of embeddings.

Notable features of our proposal include (1) the coverage of a broad band-width of use cases, (2) its firm grounding in commonly used community standards for lexical knowledge graphs, (3) the possibility to provide machine-readable metadata and machine-readable lexical data along, and even in close integration with word, lexeme, sense or concept embeddings, (4) the extension beyond fixed-size vectors as currently dominating in language technology applications, (5) and a designated vocabulary for organizing contextualized embeddings with explicit links to both their respective context and the lexical entities they refer to.

At the time of writing, one dataset (AutoExtend) has been successfully transformed from its traditional publication form and integrated with a lexical knowledge graph. However, we are still in the process of evaluating which WordNet edition the AutoExtend data actually refers to. Another dataset currently under construction is a dictionary of African sign languages, where time series information is being encoded as attestation-level embeddings.

We see our work as a building block for the development of convergencies between the Linguistic Linked Open Data and the NLP/ML community, as a conjoint modelling – or, at least, compatible levels of representation, allow to combine both technology stacks to common problems. For the NLP/ML community, machine-readable metadata and adherence to established community standards will facilitate the potential for verifying, enriching and building embeddings with or against lexical resources. For computational lexicography, closer ties with the language technology community will facilitate the uptake of machine-learning methods and their evaluation, and intensify synergies between both fields.

Acknowledgements

The research presented in this paper was in part supported by the European Horizon2020 Prêt-à-LLOD project (grant agreement no. 825182), by the project "Linked Open Dictionaries" (LiODi), funded by the German Ministry of Education and Science (BMBF), and the COST Action CA18209 European network for Web-centred linguistic data science ("NexusLinguarum").

References

[1] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[2] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994.

[3] Francis Bond, Piek Vossen, John P McCrae, and Christiane Fellbaum. Cili: the collaborative interlingual index. In *Proceedings of the Global WordNet Conference*, volume 2016, 2016.

[4] Christian Chiarcos, Maxim Ionov, Jesse de Does, Katrien Depuydt, Fahad Khan, Sander Stolk, Thierry Declerck, and John Philip McCrae. Modelling frequency and attestations for ontolex-lemon. In *Proceedings of the 2020 Globalex Workshop on Linked Lexicography*, pages 1–9, 2020.

[5] Philipp Cimiano, John P. McCrae, and Paul Buitelaar. Lexicon Model for Ontologies: Community Report, 2016.

[6] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

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

[8] Laurent Fasnacht. mmappickle: Python 3 module to store memory-mapped numpy array in pickle format. *Journal of Open Source Software*, 3(26):651, 2018.

[9] Javier D Fernández, Miguel A Martínez-Prieto, Claudio Gutiérrez, Axel Polleres, and Mario Arias. Binary rdf representation for publication and exchange (hdt). *Journal of Web Semantics*, 19:22–41, 2013.

[10] Gil Francopoulo, Monte George, Nicoletta Calzolari, Monica Monachini, Nuria Bel, Mandy Pet, and Claudia Soria. Lexical markup framework (lmf). In *International Conference on Language Resources and Evaluation-LREC 2006*, page 5, 2006.

[11] Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. Glossbert: Bert for word sense disambiguation with gloss knowledge. *arXiv preprint arXiv:1908.07245*, 2019.

[12] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, 1986.

[13] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *In Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 2–11. ACM Press, 2003.

[14] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[15] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[16] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[17] Sascha Rothe and Hinrich Schütze. Autoextend: Combining word embeddings with semantic resources. *Computational Linguistics*, 43(3):593–617, 2017.

[18] S. S Kumar, T. Wangyal, V. Saboo, and R. Srinath. Time series neural networks for real time sign language translation. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 243–248, 2018.

[19] TEI. TEI P5: Guidelines for electronic text encoding and interchange. Technical report, 2019.

[20] Gregor Wiedemann, Steffen Remus, Avi Chawla, and Chris Biemann. Does bert make any sense? interpretable word sense disambiguation with contextualized embeddings. *arXiv preprint arXiv:1909.10430*, 2019.