Word-sense Induction on a Corpus of Buddhist Sanskrit Literature

Matej Martinc[1], Andraž Pelicon[1], Senja Pollak[1], Ligeia Lugli[2]

[1]Jožef Stefan Institute, Ljubljana, Slovenia
[2] Mangalam Research Center for Buddhist Languages, Berkeley, CA, USA
E-mail: matej.martinc@ijs.si, andraz.pelicon@ijs.si, senja.pollak@ijs.si, ligeia.lugli@kcl.ac.uk

Abstract

We report on a series of word sense induction (WSI) experiments conducted on a corpus of Buddhist Sanskrit literature with an objective to introduce a degree of automation in the labour-intensive lexicographic task of matching citations for a lemma to the corresponding sense of the lemma. For this purpose, we construct a Buddhist Sanskrit WSI dataset consisting of 3,108 sentences with manually labeled sense annotations for 39 distinct lemmas. The dataset is used for training and evaluation of three transformer-based language models fine-tuned on the task of identifying intended meaning of lemmas in different contexts. The predictions produced by the models are used for clustering of lemma sentence examples into distinct lemma senses using a novel graph-based clustering solution. We evaluate how well do the obtained clusters represent the true sense distribution of new unseen lemmas not used for model training and report the best Adjusted Rand Index (ARI) score of 0.208, and how well do the clusters represent the true lemma sense distribution when the classifier is tested on new unseen sentence examples of lemmas used for model training and report the best ARI score of 0.3. In both scenarios, we outperform the baseline by a large margin.

Keywords: Buddhist Sanskrit; Word sense induction; Transformer language models

1. Introduction

Buddhist Sanskrit literature constitutes the textual foundation of Mahāyāna, one of the main branches of Buddhism, which flourished in India from around the first couple of centuries BCE to the XII century CE. The experiments reported in this paper stem from a long-standing lexicographic project aimed at creating a first corpus-based dictionary of Buddhist Sanskrit vocabulary (Lugli, 2019, 2021a). Relatively little is known about the semantic permutations that this vocabulary undergoes in different periods and text types, a corpus of relevant sources having become available only recently ((Lugli et al., 2022)). Hence, mapping word senses across various subcopora of Buddhist literature is a priority in our dictionary and, more generally, in the field of South Asian Buddhist studies. Alas, such mapping is extremely laborious. It requires close reading large quantities of citations for a given lemma. Many of these citations are extracted from highly specialised philosophical discourse and are often challenging to interpret. It took us the most part of a decade to semantically categorize a sample of just over four thousand citations that instantiate word-senses for about 130 lemmas in different genres and periods of Buddhist Sanskrit literature.

Accelerating the process of semantic categorization is clearly the key to scaling up our lexicographic endeavor and achieve a good coverage of the Buddhist Sanskrit lexicon. In this paper we report on a series of word sense induction experiments that we attempted in an effort to integrate a degree of automation in our semantic categorization workflow.

A word sense is a discrete representation of one aspect of the meaning and is context dependent. Dictionaries and lexical databases, such as WordNet (Miller, 1992), organise the entries according to different word meanings. Word Sense Disambiguation (WSD) and Word Sense Induction (WSI) are two fundamental tasks in Natural Language Processing, i.e., those of, respectively, automatically assigning meaning to words in context from a predefined sense inventory and discovering senses from text for a given input word (Navigli, 2012). Both tasks are most frequently applied to open-class words, as those are carrying most of a sentence's meaning and contain higher level of ambiguity. While for WSD the task consists of associating a word in context with its most appropriate sense from a predefined sense inventory, WSI refers to automatically identifying and grouping different senses of meanings of a word in a given textual context, without exploiting any manually sense-tagged corpus to provide a sense choice for a word in context. The output of WSI is a set of different occurrence clusters, which represent different meanings of a word. When dealing with languages with available large sense inventories, usually WSD methods are being used. On the other side, in less-resourced settings, such as in our case of Buddhist Sanskrit literature, large sense repositories are not available and therefore WSI methods are of core interest.

Therefore, the main aim of this paper is to introduce novel resources for Buddhist Sanskrit related to WSI[1], including:

- a novel word sense induction dataset for Buddhist Sanskrit containing 3108 sentences with manually labeled sense annotations (see Section 3);
- a novel graph-based WSI solution that leverages predictions produced by the transformer-based (Vaswani et al., 2017) language models fine-tuned on the binary classification task of predicting whether the target lemma in two concatenated sentences containing the lemma has the same sense or not;
- an extensive experimental evaluation of three distinct language models and two clustering algorithms, one of them being the widely used Louvain algorithm (Que et al., 2015).

The paper is structured as follows. After related work described in Section 2, we describe the data used in Section 3. Section 4 covers the training of transformer models and the novel clustering solution. Section 5 provides details on the evaluation scenarios, the baselines used and the evaluation measures. While in Section 6 we present the results of our experiments, the paper concludes with final remarks in Section 7.

2. Related work

Word sense induction and disambiguation tasks gained traction more then a decade ago, when several shared tasks on the topic were organized, the most influental being the

[1] The code for experiments is publicly available under the MIT license at https://gitlab.com/matej.martinc/buddhist-sanskrit-sense-induction.

Semeval-2010 task 14: Word sense induction and disambiguation (Manandhar et al., 2010) and the SemEval-2013 Task 13: Word Sense Induction for Graded and Non-Graded Senses (Jurgens & Klapaftis, 2013). In these challenges, one of the most common approaches was to build a word co-occurrence graph and use the relations in the graph to obtain word communities, which distinguish senses (Jurgens, 2011).

More recent approaches employ contextual embeddings (Devlin et al., 2019) for the WSI task. For example, the approach by Amrami & Goldberg (2018) is based on the intuition that occurrences of a word that share a sense, also share in-context substitutes. They use a masked language model to derive nearest word substitutes for each word and then cluster the obtained substitute vectors to derive word senses. This substitute-based approach was improved on in the study by Eyal et al. (2022). They show that the approach by Amrami & Goldberg (2018) can be adapted to efficiently and cheaply annotate all the corpus occurrences of all the words in a large vocabulary. They induce senses to a word using contextual representations from a language model and subsequently cluster them into sense clusters. More specifically, for each sentence in which the word appears, they generate k substitute tokens for the target word using a language model. Finally, they cluster all the substitutes into sense clusters. We employ their approach as one of the baselines in our work.

Another WSI method based on contextual embeddings is called PolyLM and was proposed in Ansell et al. (2021). This method combines the task of learning sense embeddings by jointly performing language modeling and word sense induction. This allows the model to utilize the advantages of contextualization at the same training step as modelling senses. PolyLM is based on two underlying assumptions about word senses: firstly, that the probability of a word occurring in a given context is equal to the sum of the probabilities of its individual senses occurring; and secondly, that for a given occurrence of a word, one of its senses tends to be much more plausible in the context than the others. Similar to the other language models, PolyLM is trained in an unsupervised manner on large corpora of unlabeled data and at inference time performs word sense induction without supervision.

Another way to tag words senses is to employ Word Sense Disambiguation (WSD), if a predefined sense inventory is available. These approaches can be roughly divided into supervised WSD and knowledge-based WSD (see Bevilacqua et al. (2021) for a recent survey). Knowledge-based approaches leverage lexical resources, including databases, such as WordNet (Miller, 1992). One of the most popular knowledge-base WSD approaches is the Lesk dictionary-based algorithm (Lesk, 1986), which also inspired one of the baseline approaches in our work. More recent vector-based approaches leverage contextualized word representations and sense embeddings to perform disambiguation (Wang & Wang, 2020). Other popular approaches leverage graph structure of knowledge graphs. A variety of graph algorithms have been employed, including random walks (Agirre et al., 2014) and Personalized PageRank (Scozzafava et al., 2020). While knowledge-base WSD (Pasini & Navigli, 2017) does not require large annotated word-to-sense corpora, they on the other hand do require a language-specific sense inventory, such as for example WordNet.

For the supervised WSD, an adequate amount of annotated data for training is required. One of the first approaches was proposed by Zhong & Ng (2010), who decided to tackle the task with an SVM-based approach. More recent studies on the other hand opted to include neural representations into the workflow. For example, several contextual embeddings based WSD approaches were tested in the scope of the SemDeep-5's Word-in-Context task

(Pilehvar & Camacho-Collados, 2019). During the task, several sense embedding systems were tested on a binary classification task of determining whether a certain "focus" word has or does not have the same sense in two concatenated sentences containing the word. The approach employing BERT performed the best.

Following Bevilacqua et al. (2021), recent supervised WSD approaches can be grouped into 1-nn vector-based ones (e.g., Wang & Wang (2020)), token tagger-based-ones (e.g., Bevilacqua & Navigli (2020) or sequence tagging-based ones (e.g., Huang et al. (2019)).

As far as we are aware, as of yet no WSI or WSD approaches have been employed for Buddhist Sanskrit.

3. Dataset

The dataset used for our experiments is derived from the data we annotated for our dictionary of Buddhist Sanskrit[2], with some notable modification. First, for this study we have considered only words for which more than 20 sentences have been manually annotated for sense. Second, we simplified our lexicographic dataset to include a single level of semantic annotation, out of three. We only use here annotations for word sense, leaving aside the more fine-grained categorization into subsenses, as well as the more general categorization into semantic fields–both of which are less closely linked to lexical context and therefore less amenable to automation than word sense. Subsenses especially have proven too complex to model due to their high number, with several words being associated to more than eight of them. Finally, in a few cases we have altered the hierarchy between senses and subsenses for this study, so that, whenever possible, senses are clearly connected to a specific lexical context. In our original lexicographic data, our priority was to convey the continuity between different senses of a word, especially between specialised and general-language uses (Lugli, 2021b). So, in our dictionary data we typically subsume terminological applications under the general-language sense from which they stem, even when the lexical contexts in which the specialized uses occur are markedly different from the general-language ones (see e.g. our dictionary sub voce "vitarka"). Given the importance of lexical context for automated word-sense-induction, we revised our dataset so that terminological uses that occur in specific contexts correspond to senses, rather than subsenses, and are therefore considered as distinct semantic categories in this study. The sense labels used in the dataset are the fruit of our lexicographic work and have been crafted to serve as English paraphrases of the senses expressed by a Sanskrit lemma.

The Buddhist Sanskrit word sense induction (WSI) dataset we used here consists of 3,108 sentences with manually labeled sense annotations for 39 distinct lemmas. The dataset statistics are presented in table 1. 26 of these lemmas have more than one sense (on average 3.3 distinct senses), while 13 are monosemous, and are only used in some of the experiments (see Section 5 for details).

The WSI dataset is used for fine-tuning and evaluation of three distinct transformer-based language models (Devlin et al., 2019), pretrained on a corpus of Buddhist Sanskrit literature.

[2] https://zenodo.org/record/7972951

	Num.lemmas	Num. sent.	Num. tokens	Average num. of senses
Monosemic	13	862	14,471	1
Polysemic	26	2,246	42,059	3.31
All	39	3,108	56,530	2.54

Table 1: The word sense induction dataset statistics.

4. Methodology

4.1 Transformer model training

In our experiments we test three distinct transformer-based language models trained on the Buddhist Sanskrit corpus described in Lugli et al. (2022). Namely, we trained two versions of the BERT model (Devlin et al., 2019), i.e. a "BERT base" model with 12 encoder layers, a hidden size of 768 and 12 self-attention heads, and a "BERT small" model with 8 encoder layers, a hidden size of 768 and 8 self-attention heads. Additionally, we also trained a smaller version of the GPT-2 model (Radford et al., 2019) with 8 encoder layers, a hidden size of 256 and 8 self-attention heads[3].

The main reason for testing of smaller models with less parameters are the overfitting issues reported in the studies by Sandhan et al. (2021) and Lugli et al. (2022), when large language models are pretrained on corpora that are magnitudes smaller than the e.g., English corpora on which these models were trained originally. In the study by Sandhan et al. (2021), where they trained a general Sanskrit model, they decided to tackle the overfitting issue by training a lighter version of BERT (a so-called ALBERT model (Lan et al., 2019)), which is a strategy that we also employ in this work in order to assess if possible improvements in performance can be obtained by employing a smaller model.

In our previous study (Lugli et al., 2022), where we tested several contextual embeddings models on the Buddhist Sanskrit corpus, we reported serious overfitting issues with a GPT-2 model[4], a model almost 10 times larger than the base version of BERT in terms of number of parameters, which resulted in very low embedding quality. For this reason, in this study we do not conduct experiments with a GPT-2 model of original size, but rather just test a much smaller version, which did not overfit on the small pretraining corpus.[5]

For language model pretraining (employing the masked language modeling objective for BERT models and autoregressive language modeling for GPT-2), we follow the regime proposed in Lugli et al. (2022). We pretrain both contextual models on the general Sanskrit corpus described in Lugli et al. (2022) for up to 200 epochs, and then on the Buddhist corpus, again for up to 200 epochs. The reason for pretraining on the general Sanskrit corpus is a considerable overlap in the vocabulary and grammar of general and Buddhist Sanskrit, which we believe the models might be able to leverage and learn some useful lexical, semantic, and grammatical information, and therefore compensate for the relatively small size of the Buddhist corpus. Same as in Lugli et al. (2022), we preprocess the corpus with the compound splitter proposed in Hellwig & Nehrdich (2018) to obtain word tokens.

[3] All these models are monolingual and were trained only on Sanskrit data.

[4] https://huggingface.co/docs/transformers/model_doc/gpt2

[5] The final model's size was determined by gradually reducing the number of encoder layers, attention heads and the embedding size until the overfitting problem has been overcome, i.e. until the perplexities the models have achieved on the train and test set were comparable.

The pretrained models are fine-tuned on a binary classification task of predicting whether the same lemma in two distinct sentences has the same sense. More specifically, for each lemma in the WSI dataset presented in Section 3, we define a set of its example sentences as L_i and build a binary classification dataset consisting of lemma sentence pairs that we obtain as a Cartesian product of L_i with itself. Note that we remove sentence pairs in which the first sentence is the same as the second sentence. We define the final dataset D as a union of sentence pairs L_i consisting of sentences s_1 and s_2 containing the same target lemma. More formally, D is defined with the following equation:

$$D = \bigcup_{i=1}^{n} (L_i \times L_i | (s_1 \in L_i) \neq (s_2 \in L_i))$$

For each sentence pair in the dataset D, we label whether the lemmas in it have the same sense or not. This dataset is used for fine-tuning and evaluation of language models.

4.2 Clustering examples into senses

The binary predictions produced by the models are used for clustering of lemma sentence examples into distinct lemma senses. We build one graph $G = (V, E)$ per lemma, comprised of a set of vertices V representing lemma sentence examples, and a set of edges $E \subseteq V \times V$, which are ordered pairs, representing connections between vertices. Vertices in the graph are connected if they contain lemma with the same sense. This allows us to build a (0,1)-adjacency matrix for each lemma, in which ones indicate whether pairs of vertices (in our case sentences) are adjacent (i.e., contain lemmas with the same sense) in the graph.

The resulting adjacency matrix is used for clustering of vertices (i.e. sentence examples containing the same target lemma) into sense clusters using a novel clustering solution, in which the rows of the matrix are used for construction of initial clusters. More specifically, in the first step, we create a different cluster containing the target vertice and its adjacent sentences for each example, resulting in n initial clusters, where n is a number of vertices in the graph. To obtain the final clusters, these initial clusters are merged by recursively combining the clusters with the largest intersection up to a predefined threshold of minimum intersection or maximum number of clusters. The threshold for minimum intersection was experimentally set to 0.8 and maximum number of clusters was set to 10, i.e., the merging of clusters with the largest intersection continues until at most 10 distinct clusters remain. The threshold of 10 was set due to the observation that very few lemmas in Buddhist Sanskrit have more than 10 distinct main senses. Note that due to a large variability in cluster sizes, the merging of clusters is based on normalized intersection that also takes into the account the number of vertices in the two clusters we potentially wish to merge. More specifically, the intersection I between two sets (clusters) of vertices S_i and S_k is normalized by dividing it with the size of the smaller cluster:

$$I = S_i \cap S_k / min(|S_i|, |S_k|) \tag{1}$$

The final step in the proposed clustering solution is to remove duplicate vertices, which appear in more than one cluster. Here we opted for a simple solution, which proved experimentally effective, and remove all duplicates but the one in the largest cluster. The

logic behind this strategy relies on a simple probability estimate that these outlier vertices, which do not fit neatly in a single cluster, have the greatest probability of belonging to the biggest cluster in a clustering.

5. Experimental setup

5.1 Evaluation scenarios

The obtained clusters, representing sense distributions for each lemma, are evaluated in two 5-fold cross-validation (CV) scenarios. All pretrained models are fine-tuned for 5 epochs on the binary classification task described in Section 4 for each fold in the cross-validation evaluation. We evaluate the performance of the models on the binary classification task using two measures, accuracy and macro-averaged F1-score. The latter was chosen in addition to accuracy due to unbalance between the two classes in the language model's test set.

In the first scenario, we test how well do the obtained clusters represent the true sense distribution of new unseen (polysemous and monosemous) lemmas not used for model training. In this scenario, we maintain a strict division between lemmas in the models' train set and lemmas in models' test set. We do not remove monosemous lemmas from the test set, in order to simulate a real life scenario of employing the model on new lemmas with unknown number of senses. We call this the "lemma division" setting. In the second scenario, we test how well do the clusters represent the true lemma sense distribution when classifier is tested on new unseen sentence examples for polysemous lemmas used for model training. Here, there is no division between lemmas in the train and test set, just a division between train and test set lemma sentence examples, since we wish to simulate a real life scenario of employing the model on new unlabeled occurrences of lemmas on which the model was trained, with known number of senses. In this scenario, we remove the monosemous lemmas from the test set, since sense induction on these lemmas is trivial for the models. We call this the "no lemma division" setting.

Note that in both scenarios the obtained train sets in the 5-fold CV evaluation are balanced, i.e., the number of sentence pairs with the same target lemma sense and the number of sentence pairs with the different target lemma sense are balanced by downsampling the majority class for each lemma. This also means that in both scenarios the models are only trained on the polysemous lemmas. On the other hand, we do not balance the test sets.

5.2 Baselines

The proposed approach is compared to three distinct baselines. To compare the novel clustering solution to a more commonly used graph-based clustering algorithm, we once again use binary predictions produced by the transformer models to build a graph $G = (V, E)$ for each lemma, comprised of a set of vertices V representing lemma sentence examples, and a set of edges $E \subseteq V \times V$. Two vertices (i.e. sentences) in the graph are again connected if they contain lemmas with the same sense. We fed the constructed graph to the popular Louvain clustering algorithm (Que et al., 2015) to obtain the final sense clusters.

The second baseline we apply only in the "no lemma division" scenario was inspired by the Lesk dictionary-based algorithm for word sense disambiguation (Lesk, 1986). More

specifically, a sentence containing a target lemma for which we wish to determine a sense, is considered as a bag of words (BOW). We calculate normalized intersection (see equation 1) between the set of words in the new sentence in the test set and all the sentences containing the same target lemma in the train set. The lemma in the test set sentence is assigned the sense of the target lemma in the train set sentence with the largest intersection. Note that this approach is only feasible in the "no lemma division scenario" and can only be employed for disambiguation of lemmas for which a set of labeled sentences already exists. We call this the "BOW intersection" approach.

The third baseline is an approach for large-scale word-sense induction by Eyal et al. (2022) described in Section 2. We re-implemented the approach from the original work but omitted the building of the inverted word index which was used to conserve space. Since in our experiment the dataset is several orders smaller in size, this step was unnecessary for our purpose. In our case, we generate the substitutes with a pretrained Buddhist Sanskrit "BERT base" language model. In each sentence, we mask the target word w and we generate the probability distribution across all the tokens in the vocabulary with the language model. We then take the k most probable tokens and treat them as the substitutes for the word w. This way we leverage the context in trying to induce senses for the target word. In our experiment we set the k to 20 experimentally.

For forming sense clusters, we first build a graph with substitutes as nodes where two nodes are connected if they represent substitutes that were being generated for the same word. We then cluster this graph using the Louvain clustering algorithm. The resulting clusters represent sense clusters. Using the Louvain algorithm allows us to not set the number of clusters prior to clustering but induce the number of clusters automatically from the data. This makes this method completely unsupervised as no sense labels nor the number of clusters for target words are needed to be known in advance. For this reason, we use it as a baseline in the "lemma division" setting.

5.3 Evaluation measures

We employ two distinct measures for evaluation of the clustering algorithm, Adjusted Rand Index (ARI) score (Hubert & Arabie, 1985) and an F1-score (Manandhar et al., 2010).

The F1-score measure for evaluation of word sense induction was first proposed in the Semeval-2010 task 14: Word sense induction and disambiguation (Manandhar et al., 2010) and was motivated by a similar evaluation measure used for information retrieval. The F-Score of a gold standard sense gs_i (denoted as $F(gs_i)$ in the equation below), is the maximum $F(gs_i, c_j)$ value attained at any cluster, where the F1-score of gs_i with respect to c_j, $F(gs_i, c_j)$, is defined as the harmonic mean of precision of class gs_i with respect to cluster c_j and recall of class gs_i with respect to cluster c_j. The F1-score of the entire clustering solution is finally defined as the weighted average of the F1-scores of each gold standard sense , where q is the number of gold standard senses and N is the total number of sentence examples for a specific lemma. More formally, the score is defined with the following equation:

$$F1 - score = \sum_{i=1}^{q} \frac{|gs_i|}{N} F(gs_i)$$

The main advantage of the F1-score evaluation is that it penalises systems that produce higher number of clusters (low recall) or lower number of clusters (low precision) than the gold standard number of senses. On the other hand, F1-score suffers from the matching problem, which results in the score not being able to evaluate the entire membership of a cluster, or by not evaluating every cluster (Rosenberg & Hirschberg, 2007), especially when gold standard distribution is very unbalanced. In this case, the F1-score tends to not consider the make-up of the clusters beyond the majority class.

For this reason, we employ an additional evaluation measure, ARI, which does not suffer from the matching problem, is equal to zero in the cases of trivial clustering, such as random clustering, or when the model produces a separate cluster for each context or a single cluster for all contexts, even in the case of uneven gold standard distribution. The measure was used for evaluation of WSI in several shared tasks (Navigli & Vannella, 2013; Panchenko et al., 2018). We adopt the ARI implementation from the scikit-learn library[6], which produces scores between 1 (when the clusterings are identical) and -0.5 (for especially discordant clusterings). ARI is based on the Rand Index (RI), which calculates a similarity score between two clusterings by looking at all pairs of samples and then counting pairs that are assigned in the same or different clusters in the predicted and gold standard clusterings.

ARI is calculated by adjusting the Rand Index for chance using the following equation:

$$ARI = \frac{RI - Expected\_RI}{max(RI) - Expected\_RI}$$

Both measures, F1-score and ARI, are calculated for each lemma. We obtain an overall score for a cross-validation fold by averaging the lemma scores. Finally, we average the scores across five cross-validation folds to obtain the overall cross validation scores. We also report the standard deviation of fold scores for both measures.

6. Results

The results the different language models achieve on the binary classification task of predicting whether the two lemmas have the same sense or not are presented in Table 2. According to both evaluation criteria, macro-averaged F1-score and accuracy, the best performing model in the "no lemma division" setting is GPT-2 small, achieving an F1-score of 69.33% and an accuracy of 72.09%. In the "lemma division" scenario, the best performing model is BERT base with an F1-score of 57.21% and an accuracy of 60.44%. The performances of all models in both scenarios are nevertheless comparable and standard deviation intervals intersect.

The results of different clustering solutions are presented in Table 3. In the "no lemma division" scenario, the best solution in terms of ARI is employing the novel clustering solution (in the Table 3 labeled as "custom clustering") on binary predictions generated by the BERT base model, with an ARI score of 0.3. While using the combination of the GPT-2 small model (which achieved the best macro-averaged F1-score and accuracy in the binary classification task) and the novel clustering solution also produces competitive

[6] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_rand_score.html

Model	F1 Macro	F1 Macro STD	Accuracy	Accuracy STD
No lemma division				
BERT base	67.94	2.97	69.23	3.37
BERT small	67.25	1.81	69.06	2.19
GPT-2 small	**69.33**	1.02	**72.09**	1.19
Lemma division				
BERT base	**57.21**	5.55	**60.44**	6.21
BERT small	56.47	3.44	60.09	4.93
GPT-2 small	55.20	3.17	59.71	5.85

Table 2: The results of different language models on the binary classification task.

Approach	ARI	ARI STD	F1	F1 STD
No lemma division				
BERT base + Custom clustering	**0.300**	0.034	76.10	0.58
BERT small + Custom clustering	0.217	0.041	73.96	1.58
GPT-2 small + Custom clustering	0.286	0.035	75.74	0.65
BERT base + Louvain	0.271	0.039	73.60	1.24
BERT small + Louvain	0.205	0.048	70.84	2.25
GPT-2 small + Louvain	0.258	0.043	74.28	2.13
BOW intersection	0.254	0.042	**76.78**	1.68
Lemma division				
BERT base + Custom clustering	0.099	0.026	79.78	4.56
BERT small + Custom clustering	0.116	0.029	79.94	4.38
GPT-2 small + Custom clustering	**0.208**	0.159	**80.36**	4.03
BERT base + Louvain	0.041	0.026	61.58	2.37
BERT small + Louvain	0.055	0.022	65.04	4.66
GPT-2 small + Louvain	0.023	0.022	69.14	2.56
Eyal et al. (2022)	0.024	0.010	35.50	1.10

Table 3: The results of different clustering solutions.

results in terms of ARI, employing the novel clustering solution on binary predictions produced by the BERT small model surprisingly leads to a much worse performance in terms of ARI. This finding is interesting since all the models achieved comparable performance on the binary classification task, therefore we expected that the clustering results would also be competitive. Using the Louvain clustering, we achieve lower ARI and F1-scores than using the custom clustering no matter the model we use for binary predictions. Again, employing the Louvain algorithm on binary predictions produced by the BERT small model leads to much worse results in terms of ARI than if two other models are used.

In terms of the F1-score, the non-neural BOW intersection baseline achieves the best performance of 76,78%. Using the combination of BERT base or GPT-2 small and custom clustering is also a competitive strategy, leading to F1-scores around 76%. We believe that the best performance of the BOW intersection baseline in terms of F1-score is to some extent caused by the unbalanced distribution of senses in the gold standard distribution and the fact that the F1-score tends to not consider the make-up of the clusters beyond the majority class due to the matching problem. Nevertheless, since the BOW intersection

Lemma	Key Word in Context (KWIC)	Translation	Sense	Assigned cluster
nāman	niṣikte nāma rūpe tu ṣaḍ āyatana saṃbhavaḥ / ṣaḍ āyatanam āgamya saṃsparśaḥ saṃpravartate //	When name and form develop, the six senses emerge. In dependence upon the six senses, impact actually occurs. [Batchelor]	nāma-rūpa (one of the twelve nidānas)	1
nāman	[...] yad idam a vidyā pratyayāḥ saṃskārāḥ saṃskāra pratyayaṃ vijñānaṃ vijñāna pratyayaṃ nāma rūpaṃ nāma rūpa pratyayaṃ [...]	translation not available	nāma-rūpa (one of the twelve nidānas)	2
nāman	tadyathāpi nāma subhūte ratn ārthikaḥ puruṣo mahāsamudraṃ dṛṣṭvā n āvagāheta /	Just as if a person who desires jewels would not look for them in the great ocean, [...]. [Conze 235]	namely	3
nāman	bhoḥ puruṣa kas tav āsyāṃ upary anunayo yan nāma madīyām ājñāṃ vilaṅghya n ecchasy enāṃ praghātayituṃ [...]	Man! What regard do you have for her that, violating my order, you do not wish to kill her? [Rajapatirana 19]	namely	3
nāman	tadyathāpi nāma ānanda rājā cakravartīṃ prāsādāt prāsādaṃ saṃkrāmet /	A universal monarch can pass from palace to palace, [...]. [Conze 366]	namely	3
nāman	tasya parama siddha yātratvāt supāraga ity eva nāma babhūva /	His voyages proved so extremely successful that he came to be called Supāraga. [Khoroche 96]	name/word	4
nāman	asyām ānanda mathurāyāṃ mama varṣa śata parinirvṛtasya gupto nāma gāndhiko bhaviṣyati /	Ananda, right here in Mathurā, one hundred years after my parinirvāṇa, there will be a perfumer named Gupta. [Strong 174]	name/word	4
nāman	tasya vistareṇa jātimahaṃ kṛtvā pṛcchati kiṃ kumārasya bhavatu nāma /	When the prince's full birth festival was being celebrated, she was asked what his name should be. [Strong 205]	name/word	4
nāman	paśy ājit aika sattvam api nām otsāhayitv eyat puṇyaṃ prasavati /	Mark, Agita, how much good is produced by one's inciting were it but a single creature; [Kern 333]	indeed/really/actually	4
nāman	arthibhiḥ prīta hṛdayaiḥ kīrtyamānam itas tataḥ / tyāga śaury onnataṃ nāma tasya vyāpa diśo daśa //	"[1] His petitioners were well-contented and praised him far and wide, so that the name he earned for his largesse spread to every corner of the earth. [Khoroche 22]	name/word	4

Table 4: Word sense induction examples in the "no lemma division" setting for lemma *nāman* with four distinct labeled senses, when BERT base and custom clustering is employed. In KWIC examples, and tags are used for denoting the target lemma and / (daṇḍa) for punctuation.

baseline also offers solid performance in terms of ARI (0.254), and since it does not require any additional cluster mapping[7], this approach seems like a viable option, especially since it is extremely fast and requires very few computational resources.

In the "lemma division" setting, the usage of custom clustering tends to outperform all the baseline approaches by a large margin according to both evaluation criteria. By far the best ARI score of 0.208 is achieved if we use custom clustering on the binary predictions produced by the GPT-2 small model. In this setting, the standard deviation between folds in the 5 fold CV setting is nevertheless very large, 0.159. In fact, the ARI score across folds varied between 0.477 and 0.029, which means that the score very much depends on which lemmas are in the train set, when GPT-2 small model is used for production of binary predictions. This indicates that the model might have issues finding general rules that can be applied for sense disambiguation on different lemmas and rather relies on a set of features that only work for some lemmas.

[7] While the BOW intersection baseline works as a word sense disambiguation approach by assigning target lemmas in new sentences predefined senses, the other approaches work as word sense induction strategies, producing clustering distributions without labeled clusters. While the latter approaches are useful if all word senses for a specific target lemma are not known in advance, an additional cluster mapping step, in which the produced unlabeled clusters are mapped to the actual lemma senses is nevertheless required in order to obtain actual senses.

The usage of BERT base or BERT small models leads to more consistent ARI scores across different folds of around 0.1. This means that there is a substantial drop in terms of ARI, if we compare the "lemma division" approach to the "no lemma division" approach, which suggests that all transformer models (not just the GPT-2 small model) have issues in finding general rules that can be applied for sense disambiguation on different lemmas. Most likely this is due to the limited size of the fine-tuning dataset, which only contains 39 different lemmas.

In terms of the F1-score, all approaches based on custom clustering achieve comparable and very competitive scores around 80%. Again, we believe that this is partially caused by the matching problem of the evaluation score and unbalanced distribution of senses in the gold standard distribution.

Examples of word sense induction for lemma *nāman* in the "no lemma division" setting when BERT base and custom clustering is employed are presented in Table 4. Note how the sentence examples containing lemmas with majority senses ("namely" and "name/word") tend to be clustered correctly, while the clustering perform worse for examples containing lemmas with minority senses ("indeed/really/actually" and "nāma-rūpa (one of the twelve nidānas)").

7. Conclusion

In the paper, we released the first word sense induction dataset and proposed the first WSI approach employed for Buddhist Sanskrit, with an intention to automate the time and labor intensive lexicographic task of assigning senses to target lemmas in sentences. The approach relies on pretrained transformer language models fine-tuned on a binary classification task of predicting whether two identical target lemmas in two sentences have the same sense or not. The produced predictions are then used in a novel graph-based clustering solution.

While the proposed approach outperforms several WSI baselines in terms of ARI, we do observe several potential problems with the method, which will need to be thoroughly addressed before it can be fully integrated in a lexicographic pipeline for Buddhist Sanskrit. First, the large difference in performance between the two tested approaches, the "lemma division" approach and the "no lemma division" approach, indicates that transformer models tend to rely on lemma specific features during binary classification and fail to find general contextual features to distinguish between senses. Another indication of that is the standard deviation between folds in the 5 fold CV setting in the "lemma division" setting, when the best performing GPT-2 small model is used. The latter suggests that the selection of lemmas, on which the model is trained, is important. We believe that both of these problems could be resolved by a larger training dataset in terms of both sentence examples for a specific lemma and number of different lemmas in the dataset. The construction of such bigger training dataset will be the object of future work, but it seems likely that only the number of different lemmas included in the data will increase substantially, as lexicographers will in any case progressively annotate sentences for more lemma as they expand the dictionary. By contrast, expanding the number of sentences annotated for each lemma may prove difficult to align with lexicographic goals, since manual annotation is extremely laborious and WSI is needed to reduce the amount of manual annotation required for dictionary development.

When it comes to the evaluation scores, we believe that the F1-score is not appropriate for evaluation in our setting, because the unbalanced classes resulting from the above-mentioned matching problem interfere with the score's ability to evaluate entire membership of the cluster, especially in scenarios where a prevailing gold standard majority cluster is accompanied by several smaller clusters. Since the score is calculated as the weighted average of the F1-scores of each gold standard cluster, in such scenarios the memberships of smaller clusters are neglected due to relatively small weights. In our case, this leads to a relatively small differences between different approaches in terms of F1-score (this was especially the case in the "no lemma division" scenario), since all approaches were able to assign membership to a majority cluster to a reasonably good degree, since this is the easiest part of the task. On the other hand, there were significant differences between different approaches when it comes to successfully assigning membership to minority clusters, and these were not captured by the F1-score. While the ARI score tends to do better in this respect, we will nevertheless explore other evaluation scores in future work, in order to try to improve our evaluation scenario even further.

8. Acknowledgements

This work was funded by a NEH Digital Advancement Grant level 2 (HAA-277246-21), while the creation of the Buddhist Sanskrit Corpus was partly funded by the British Academy (NF161436) and the Khyentse Foundation. We also acknowledge the Slovenian Research Agency core programme Knowledge technologies P2-0103. Finally, we would like to thank Luis Quiñones for his contribution to the creation of the evaluation dataset.

9. References

Agirre, E., de Lacalle, O.L. & Soroa, A. (2014). Random Walks for Knowledge-Based Word Sense Disambiguation. *Computational Linguistics*, 40(1), pp. 57–84. URL https://aclanthology.org/J14-1003.

Amrami, A. & Goldberg, Y. (2018). Word Sense Induction with Neural biLM and Symmetric Patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 4860–4867. URL https://aclanthology.org/D18-1523.

Ansell, A., Bravo-Marquez, F. & Pfahringer, B. (2021). PolyLM: Learning about Polysemy through Language Modeling. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, pp. 563–574. URL https://aclanthology.org/2021.eacl-main.45.

Bevilacqua, M. & Navigli, R. (2020). Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. pp. 2854–2864.

Bevilacqua, M., Pasini, T., Raganato, A. & Navigli, R. (2021). Recent Trends in Word Sense Disambiguation: A Survey. In *International Joint Conference on Artificial Intelligence*.

Devlin, J., Chang, M.W., Lee, K. & Toutanova, K. (2019). 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 Linguis-*

tics: Human Language Technologies, Volume 1 (Long and Short Papers). Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186.

Eyal, M., Sadde, S., Taub-Tabib, H. & Goldberg, Y. (2022). Large Scale Substitution-based Word Sense Induction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, pp. 4738–4752. URL https://aclanthology.org/2022.acl-long.325.

Hellwig, O. & Nehrdich, S. (2018). Sanskrit word segmentation using character-level recurrent and convolutional neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. pp. 2754–2763.

Huang, L., Sun, C., Qiu, X. & Huang, X. (2019). GlossBERT: BERT for Word Sense Disambiguation with Gloss Knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 3509–3514.

Hubert, L. & Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2, pp. 193–218.

Jurgens, D. (2011). Word sense induction by community detection. In *Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing*. pp. 24–28.

Jurgens, D. & Klapaftis, I. (2013). SemEval-2013 Task 13: Word Sense Induction for Graded and Non-Graded Senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Atlanta, Georgia, USA: Association for Computational Linguistics, pp. 290–299. URL https://aclanthology.org/S13-2049.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P. & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Lesk, M. (1986). 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*. pp. 24–26.

Lugli, L. (2019). Smart lexicography for under-resourced languages: lessons learned from Sanskrit and Tibetan. In *Smart Lexicography: eLex 2019*. pp. 198–212.

Lugli, L. (2021a). Dictionaries as collections of data stories: an alternative post-editing model for historical corpus lexicography. In *Post-Editing Lexicography: eLex 2021*. pp. 216–231.

Lugli, L. (2021b). Words or terms? Models of terminology and the translation of Buddhist Sanskrit vocabulary. In A. Collett (ed.) *Buddhism and Translation: Historical and Contextual Perspectives*. pp. 149–172.

Lugli, L., Martinc, M., Pelicon, A. & Pollak, S. (2022). Embeddings models for Buddhist Sanskrit. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. pp. 3861–3871.

Manandhar, S., Klapaftis, I., Dligach, D. & Pradhan, S. (2010). Semeval-2010 task 14: Word sense induction &disambiguation. In *Proceedings of the 5th international workshop on semantic evaluation*. pp. 63–68.

Miller, G.A. (1992). WordNet: A Lexical Database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*. URL https://aclanthology.org/H92-1116.

Navigli, R. (2012). A Quick Tour of Word Sense Disambiguation, Induction and Related Approaches. In M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser & G. Turán

(eds.) *SOFSEM 2012: Theory and Practice of Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 115–129.

Navigli, R. & Vannella, D. (2013). Semeval-2013 task 11: Word sense induction and disambiguation within an end-user application. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. pp. 193–201.

Panchenko, A., Lopukhina, A., Ustalov, D., Lopukhin, K., Arefyev, N., Leontyev, A. & Loukachevitch, N. (2018). RUSSE'2018: a shared task on word sense induction for the Russian Language. *arXiv preprint arXiv:1803.05795*.

Pasini, T. & Navigli, R. (2017). Train-O-Matic: Large-Scale Supervised Word Sense Disambiguation in Multiple Languages without Manual Training Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 78–88. URL https://aclanthology.org/D17-1008.

Pilehvar, M.T. & Camacho-Collados, J. (2019). WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 1267–1273. URL https://aclanthology.org/N19-1128.

Que, X., Checconi, F., Petrini, F. & Gunnels, J.A. (2015). Scalable community detection with the louvain algorithm. In *2015 IEEE International Parallel and Distributed Processing Symposium*. IEEE, pp. 28–37.

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

Rosenberg, A. & Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. pp. 410–420.

Sandhan, J., Adideva, O., Komal, D., Behera, L. & Goyal, P. (2021). Evaluating Neural Word Embeddings for Sanskrit. *arXiv preprint arXiv:2104.00270*.

Scozzafava, F., Maru, M., Brignone, F., Torrisi, G. & Navigli, R. (2020). Personalized PageRank with Syntagmatic Information for Multilingual Word Sense Disambiguation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, pp. 37–46. URL https://aclanthology.org/2020.acl-demos.6.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*. pp. 5998–6008.

Wang, M. & Wang, Y. (2020). A Synset Relation-enhanced Framework with a Try-again Mechanism for Word Sense Disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 6229–6240. URL https://aclanthology.org/2020.emnlp-main.504.

Zhong, Z. & Ng, H.T. (2010). It Makes Sense: A Wide-Coverage Word Sense Disambiguation System for Free Text. In *Proceedings of the ACL 2010 System Demonstrations*. Uppsala, Sweden: Association for Computational Linguistics, pp. 78–83. URL https://aclanthology.org/P10-4014.