MWE as WSD: Solving Multiword Expression Identification with Word Sense Disambiguation

***Joshua Tanner**
Mantra Inc.
josh@mantra.co.jp

***Jacob Hoffman**
Project Ronin
jacob@projectronin.com

Abstract

Recent approaches to word sense disambiguation (WSD) utilize encodings of the sense gloss (definition), in addition to the input context, to improve performance. In this work we demonstrate that this approach can be adapted for use in multiword expression (MWE) identification by training models which use gloss and context information to filter MWE candidates produced by a rule-based extraction pipeline. Our approach substantially improves precision, outperforming the state-of-the-art in MWE identification on the DiMSUM dataset by up to 1.9 F1 points and achieving competitive results on the PARSEME 1.1 English dataset. Our models also retain most of their WSD performance, showing that a single model can be used for both tasks. Finally, building on similar approaches using Bi-encoders for WSD, we introduce a novel Poly-encoder architecture which improves MWE identification performance.

1 Introduction

Word sense disambiguation (WSD), the task of predicting the appropriate sense for a word in context, and multiword expression (MWE) identification, the task of identifying MWEs in a body of text, both deal with determining the meaning of words in context (Maru et al., 2022; Constant et al., 2017). They have traditionally been treated as separate tasks, but this is potentially disadvantageous as WSD performed on words which are part of unrecognized MWEs cannot produce correct meanings, and the meanings of polysemous MWEs are ambiguous even after identification. For example, the sentence "She inherited a fortune after her grandfather kicked the bucket" tells us that someone's grandfather has died, but we would not expect to find meanings associated with death in the sense inventories of either *kick* or *bucket*. WSD cannot capture the meanings of these words in context unless the relevant MWE is identified first. However, like many MWEs, *kick the bucket* can have a literal, non-compositional meaning as in "He kicked the bucket down the hill," so we also cannot indiscriminately mark all combinations of words in known MWEs as MWEs. MWEs can also have multiple possible senses in the same way words can: *break up* can refer both to objects physically breaking apart and romantic relationships ending, so even in cases where it is correctly identified as a MWE its meaning is ambiguous without WSD. Identifying the meanings of all words in a sentence requires solving these tasks together.

WSD and MWE identification can be used in preprocessing to improve performance of downstream tasks such as machine translation or information extraction (Zaninello and Birch, 2020; Song et al., 2021; Barba et al., 2021a). They also have more direct applications in helping language learners – for whom MWEs are particularly challenging (Christiansen and Arnon, 2017; Pulido, 2022) – understand the meaning of words or MWEs in context.

In this paper, we propose a system that tackles these tasks together, using a MWE lexicon and rule-based pipeline to identify MWE candidates and a trainable model to both perform WSD and filter MWE candidates. Our model is a modified Poly-encoder (Humeau et al., 2020), a natural extension of previous work using Bi-encoders for WSD (Blevins and Zettlemoyer, 2020; Kohli, 2021). Utilizing gloss information[1] allows our model to consider the meaning of MWEs and filter out candidates where the constituents of a MWE are present but the MWE meaning does not fit the context, such as the aforementioned literal usage of *kick the bucket*. Our method improves precision and achieves state-of-the-art F1 for MWE identification

*Both authors contributed equally to this work.

[1]For example, in "The couple **broke up** amicably", the gloss, or definition, of the sense of **break up** is "discontinue an association or relation; go different ways." (Miller, 1995)

Findings of the Association for Computational Linguistics: EMNLP 2023, pages 181–193

on the DiMSUM dataset (Schneider et al., 2016) and competitive performance on the PARSEME 1.1 English data (Ramisch et al., 2018). To the best of our knowledge, this work is the first to use glosses as a resource for MWE identification. Our contributions are summarized as follows:

- We present a system which solves MWE identification and WSD together, achieving state-of-the-art results for MWE identification on DiMSUM and only 6% less F1 for WSD than an equivalent single-task model
- We propose a novel Poly-encoder architecture which outperforms standard Poly-encoders on both tasks, and Bi-encoders on PARSEME MWE identification
- We explore why our system performs well and where it falls short through ablations and a detailed error analysis with examples

We make all of our code, models and data public[2].

2 Related Work

2.1 Word Sense Disambiguation

Until the last few years, most approaches to WSD treated senses only as one of many possible labels in a classification task. This formulation limits the information available to the model about each sense to only what is learnable from the training data, and can lead to poor performance on rare or unseen senses. To mitigate these problems, recent approaches have improved performance by incorporating sense glosses (Blevins and Zettlemoyer, 2020; Barba et al., 2021a; Zhang et al., 2022).

Our work is inspired by this methodology and utilizes gloss information to improve MWE identification. In particular, Blevins and Zettlemoyer (2020) demonstrate that a simple Bi-encoder model consisting of two BERT (Devlin et al., 2019) models can achieve competitive WSD performance, with Kohli (2021) improving Bi-encoder training for WSD and Song et al. (2021) achieving further performance gains through improved sense representations. Bi-encoder models are also particularly efficient at inference time because gloss representations can be computed in advance and cached.

2.2 Poly-encoders

The Poly-encoder architecture was proposed by Humeau et al. (2020) as a middle ground between Bi-encoders and Cross-encoders (which jointly encode all possible input pairs), retaining the speed advantage of the Bi-encoder, but allowing information to flow between the two encoder outputs like the Cross-encoder. It can be used in place of a Bi-encoder in tasks such as information retrieval (Li et al., 2022) text reranking (Kim et al., 2022), or in our case MWE identification and WSD.

2.3 Multiword Expression Identification

Precisely defining what constitutes a MWE has proven to be difficult (Maziarz et al., 2015), but they can be broadly defined as groupings of words whose meaning is not entirely composed of the meanings of included words (Sag et al., 2002; Baldwin and Kim, 2010). This includes idioms such as *a taste of one's own medicine*, verb-particle constructions such as *break up* or *run down*, compound nouns such as *bus stop*, and any other grouping of words with non-compositional semantics. In fact, a significant portion of noun MWEs are named entities (Savary et al., 2019).

The task of MWE identification is locating these MWEs in a given body of text. Common approaches to solving MWE identification include rule-based systems (Foufi et al., 2017; Pasquer et al., 2020), CRF-based systems (Liu et al., 2021), and token tagging systems (Rohanian et al., 2019). Rule-based systems remain competitive with neural models in this task, and many systems including ours use MWE lexicons in order to identify MWEs, which Savary et al. (2019) argue are critical to making progress in MWE identification. Kurfalı and Östling (2020) and Kanclerz and Piasecki (2022) are similar to our work in that they frame the task of MWE identification as a classification problem, although neither use gloss information.

Among all the types of MWEs, verbal MWEs are particularly difficult to identify due to their surface variability — constituents can be conjugated or separated so that they become discontinuous (Pasquer et al., 2020). Much work on verbal MWE identification, especially in languages other than English, has been done as part of recent iterations of the PARSEME shared task (Ramisch et al., 2018).

3 Methodology

In this section, we explain how our models perform MWE identification and WSD, and how our MWE identification pipeline works.

[2]Code, data and links to models available at https://github.com/Mindful/MWEasWSD

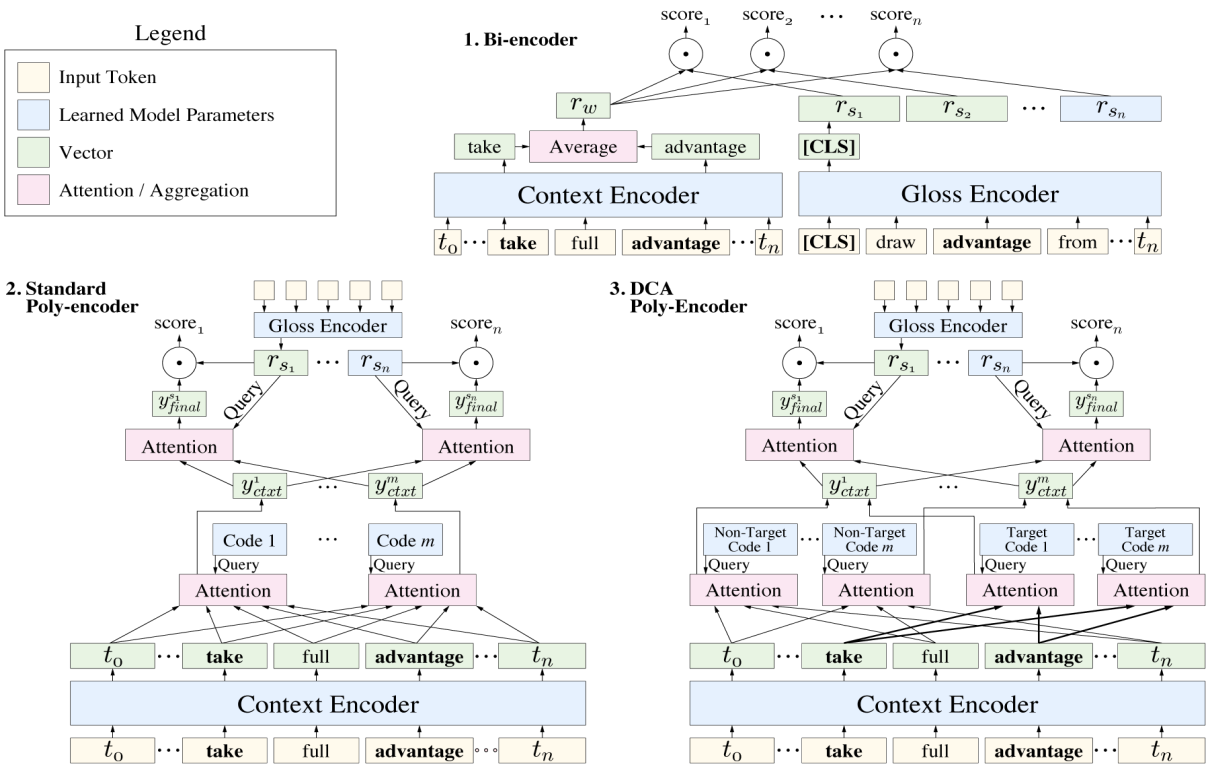

Figure 1: Each model scoring the MWE **take advantage**. "Draw advantage from" is the gloss for one possible sense. The gloss encoder produces sense representations r_s using the [CLS] embedding in all models. The MWE representation r_w is an average of constituents for the Bi-encoder and the combination of attention for each code for the Poly-encoder. The DCA Poly-encoder learns separate codes for target and non-target tokens, allowing it to attend differently to the MWE and surrounding context. Scores are the similarity between r_s and r_w computed as the dot product, and the model predicts the sense with the highest score.

3.1 Bi-encoder

Bi-encoders for WSD, as defined by Blevins and Zettlemoyer (2020), consist of two BERT (Devlin et al., 2019) models: a **context encoder** T_c and **gloss encoder** T_g, which embed the context and sense glosses into the same embedding space. Given an input sentence $c = (w_0, ...w_n)$ containing the target words to disambiguate, we first tokenize it and use the **context encoder** to produce representations for each token. Because tokenization may break words or MWEs up into multiple subwords, word or MWE representations r_w are computed as an average of all included subwords.

$$T_c(c) = t_0, ...t_n$$
$$r_w = \frac{1}{|w|} \sum_{t \in w} t$$

Then, for each target word or MWE, the **gloss encoder** produces a sense representation r_s for each possible sense by encoding its gloss and taking the [CLS] token embedding.

$$r_s = T_g(g_s)[0]$$

Scores corresponding to possible senses for each target word are computed as the dot product similarity of the word and sense representations, and the model predicts the highest scoring sense.

$$\phi(w, s_i) = r_w \cdot r_{s_i}$$
$$pred(w) = \arg\max_{s_i} \phi(w, s_i) : s_i \in S_w$$

3.2 Poly-encoder

Like the Bi-encoder, the Poly-encoder has a **context encoder** T_c for target word contexts and a **gloss encoder** T_g for glosses. There is also a new set of parameters that Humeau et al. (2020) refer to as **code embeddings**, Q. These codes are used as queries to extract information from context representations produced by the **context encoder**. The inputs to the Poly-encoder are the same as to the Bi-encoder, sense representations r_s are computed identically, and predictions are still the highest scoring sense. However, senses are scored differently.

We take the last hidden state of the **context encoder** as the context representation $r_c = T_c(c)$, which we use along with the code embeddings $Q = (q_1, ..., q_m)$ in the first dot-product attention step (*code context attention*) of the Poly-encoder.

We use a different set of embeddings for single words and MWEs. The number of embeddings, m, is a hyperparameter and their dimensionality is the same as the encoders' hidden sizes. The context representation r_c is used as both keys and values in this dot-product attention module, yielding a *code attended context* Y_{ctxt}. The representation a code q_i extracts is as follows:

$$(w_0^{q_i}, ..., w_n^{q_i}) = softmax(q_i \cdot r_{c_1}, ...q_i \cdot r_{c_n})$$
$$y_{ctxt}^i = \sum_{j=1}^{n} w_j^{q_i} r_{c_j}$$

Sense representations r_s are then used as queries and the code-attended context representations Y_{ctxt} are used as keys and values in a final dot-product attention module, yielding a *gloss attended code-context*. For a given sense s of a word or MWE:

$$(w_1, ..., w_m) = softmax(r_s \cdot y_{ctxt}^1, ..., r_s \cdot y_{ctxt}^m)$$
$$y_{final}^s = \sum_{i=1}^{m} w_i y_{ctxt}^i$$

We then take the dot product of the gloss attended code-context y_{final} and each gloss embedding $r_{s_0}, ...r_{s_k}$, yielding a score for each gloss: $\phi(w, s_i) = y_{final} \cdot r_{s_i}$.

3.3 Distinct Codes Attention

Since the Poly-encoder was originally designed to compute *sentence* representations, it contains no mechanism for explicitly focusing on a specific set of target words/subwords. To address this problem, we propose a variation of the Poly-encoder which we call "distinct codes attention" (DCA). We change the *code context attention* step of the Poly-encoder so that it can attend differently to target words and the surrounding context, using two sets of code embeddings: one set for target words, Q_t and one set for non-target words Q_{nt}. Since we also maintain different code embeddings for single words and MWEs, this gives us a total of four sets of code embeddings.

In the first attention module, *code-context attention*, we construct two key matrices, one to be used with the target code queries Q_t and one to be used with the nontarget code queries Q_{nt}. First we create two masks which pick out target or nontarget subwords: the target mask M_t, which is 1 at the indices of target subwords and 0 otherwise, and the nontarget mask M_{nt} which is the opposite. We then multiply each mask by the encoded context r_c to get target and nontarget key matrices $K_t = M_t r_c$ and $K_{nt} = M_{nt} r_c$. Next we compute target and nontarget query results (QK^T) and add them.

$$QK^T = Q_t K_t^T + Q_{nt} K_{nt}^T$$

Finally, we softmax and multiply QK^T by the encoded context r_c to yield the *code attended context*, $Y_{ctxt} = softmax(QK^T)(r_c)$. The *gloss attended code-context* and final scores are then computed identically to the standard Poly-encoder.

3.4 MWE Identification Pipeline

We use a rule-based pipeline inspired by Kulkarni and Finlayson (2011) for MWE identification. First, we compute initial candidates as all combinations of words in a sentence whose lemmas correspond to a MWE in our lexicon. That is, any group of words that when lemmatized corresponds to a known MWE, regardless of order or location in the sentence, is a candidate. This ensures we rarely miss known MWEs, but also produces many false positives, such as: **in that** in "**That** was back **in** 1954, 55 years ago."

Next, we filter the candidates by removing MWEs which are out of order or too gappy (>3 words in between constituents), and optionally by discarding MWE candidates judged to be incorrect by our DCA Poly-encoder (or other) model. We refer to the combination of rule-based extraction and filters with no model as the *rule-based pipeline*. Since the model is applied as a final filter after extraction and the other filters, it can only improve precision. While the heuristic filters involving order and gappyness exclude some valid MWEs as well, they empirically improved performance on development data, and the majority of exclusions made by these filters are correctly removing false positives from candidate generation as can be seen in Table 1. Note that many candidates excluded by one filter would also be excluded by another filter later in the pipeline.

	PARSEME		**DiMSUM**	
	TN	FN	TN	FN
OrderedOnly	1005	29	427	23
MaxGappiness	1549	52	655	49

Table 1: Tokens excluded by rule-based filters. True negatives represent correct exclusions (I.E. false positive candidates), and false negatives incorrect exclusions.

In cases of overlap between candidates remaining after filtering, we use only the candidate judged to be most likely by our model (or least gappy, in pipelines without models).

3.4.1 Model Filter

Because all of our MWE candidates correpond to words (and consequently subwords) in the input sentence, we can produce a representation r_w for each MWE candidate, along with scores for each of their possible senses, the same way we do for words. However, since no MWE has a sense corresponding to the case where that candidate is a false positive, we define a special sense s_n representing the case where the candidate is not a MWE. Since s_n has no gloss, we cannot use the **gloss encoder** to compute a representation for it, and instead make this representation a learnable parameter matrix r_{s_n}, with the same dimensionality as the model's hidden size. This can then be used in our model's scoring functions to compute a score for the candidate not being a MWE. When using a model to filter, we exclude any MWE candidates whose highest scoring sense is the "not a MWE" sense s_n, retaining only candidates for which the below is true:

$$\exists s_i \in S_w \; \phi(w, s_i) > \phi(w, s_n)$$

Note that since this filtering process involves computing scores for all possible senses, it also effectively performs WSD on any polysemous MWEs.

4 Experimental Setup

4.1 Lexicon

We use WordNet (Miller, 1995) as our MWE lexicon for all experiments, treating every entry including the character "_" as a MWE. All sense glosses are taken from WordNet 3.0.

4.2 Training Data

We train our models on SemCor (Miller et al., 1993), a WSD dataset containing a total of 226,036 examples annotated with senses from WordNet. In order to make the data usable for MWE identification in addition to WSD, we preprocess it in the following ways. First, we explicitly mark any words whose lemma includes the character "_" as MWEs such that during training the possible labels for these MWEs also include the "not a MWE" sense. Since some discontiguous MWEs in SemCor are labeled only on a subset of the included words, we add stranded constituents to their parent MWE by attaching nearby words whose lemmas match constituents missing from the labeled MWE[3]. Finally, because SemCor contains no labeled negative examples of MWEs — instances where the constituent words of a MWE are all present but their meaning in context does not match any of the MWE senses — we add these ourselves. We generate synthetic negative examples using the rule-based pipeline with its filters inverted to mark combinations of words whose lemmas correspond to a known MWE but are out of order or very gappy as negative examples whose gold label is the "not a MWE" sense. We randomly add negative examples in this fashion until they account for just over 50% of the MWE examples in the training data.

To mitigate the risk of the model learning only the heuristics used to generate these synthetic negatives, we also manually annotate a small number of examples. We do this by running the rule-based pipeline (Section 3.4) on the SemCor data and annotating output MWEs with their appropriate sense from WordNet or the "not a MWE", sense based on context. Because we exclude words already marked as MWEs and many MWEs in SemCor have already been annotated, >50% of the newly annotated examples are negative.

	Pos MWE	Neg MWE
SemCor	12409	0
+Annotation	12907	658
+Synthetic Negatives	12907	14688

Table 2: SemCor after each addition of data

Context	MWE	Type
What effort **do** you **make** to assess ...	**make do**	Synthetic Negative
..your **in** plant feeding **operation**?	**in operation**	Annotated Negative
...**works** full-time **on** some other assignment?	**work on**	Annotated Positive

Table 3: Examples of each annotation type

4.2.1 Fine-tuning Data

After training on SemCor, we fine-tune on the MWE identification data in DiMSUM/PARSEME. We use any labeled examples of MWEs which are

[3]For example, in "Are they encouraged to take full legal advantage of these benefits?" (ID d000.s015), the verb *take* is correctly labeled as the MWE *take_advantage*, but *advantage* is not labeled as being part of any MWE, so we attach it.

in our lexicon as positive examples, and then run our full pipeline (rules+model filter) on the data and take incorrect outputs as negative examples. This means that all the negative training examples used in fine-tuning are false positives from the model itself, allowing the model to learn from its mistakes. Because PARSEME and DiMSUM are not annotated with sense information (only a binary labeling of MWE or not), we use the first sense from WordNet as the gold label for positive examples when fine-tuning. For both datasets, we use 10% of the training data as our development set.

4.3 Training

Like Blevins and Zettlemoyer (2020), we train with cross-entropy loss. The difference is that for MWEs, there is one additional possible label representing the "not a MWE" case. Given a word or MWE w, its gold sense g_s, and $|S_w| = j$ possible senses in the lexicon, this formalizes to:

$$\mathcal{L}(w, g_s) = -\phi(w, g_s) + \log \sum_{x \in X} \exp(\phi(w, x))$$

$$X = \begin{cases} \{s_0, ...s_j, n\} & \text{if MWE} \\ \{s_0, ...s_j\} & \text{otherwise} \end{cases}$$

We train for 15 epochs on SemCor and three epochs for fine-tuning, computing F1 on the WSD and MWE identification dev sets once per epoch and use the best performing model as our final model. Batch size and other hyperparameters such as learning rate were determined by hyperparameter search. Further implementation and training details can be found in Appendix A.

4.4 MWE Identification Evaluation

We evaluate our system on the English section of the PARSEME 1.1 Shared Task (Ramisch et al., 2018) and the DiMSUM dataset (Schneider et al., 2016). We do not evaluate on STREUSLE (Schneider et al., 2018) as it requires predicting lexical categories and supersenses[4], while our system predicts only the presence or absence of MWEs. To measure WSD performance, we use the evaluation framework established by Raganato et al. (2017) and evaluate on the English all-words task.

4.4.1 PARSEME 1.1

The PARSEME data focuses on verbal MWEs, containing 3471 sentences in the training set and 3965 in test. Because the data contains only verbal MWEs, when evaluating on PARSEME we limit the output of our pipeline to verbal MWEs.

4.4.2 DiMSUM

The DiMSUM data consists of online reviews, tweets and TED Talks which have been annotated with MWEs and other information. There are 4799 sentences in the training set, and 1000 in the test set. Because noun phrases are marked as MWEs in DiMSUM, when evaluating on DiMSUM our pipeline also marks consecutive nouns as MWEs.

4.5 WSD Evaluation

Following standard practice, we use the SemEval-2007 dataset (Pradhan et al., 2007) as our dev set, holding out the remaining Senseval-02, Senseval-03, SemEval-2013, and SemEval-2015, as test sets (Palmer et al., 2001; Snyder and Palmer, 2004; Navigli et al., 2013; Moro and Navigli, 2015).

5 Results

Table 4 shows MWE identification performance for the rule-based pipeline (Section 3.4), and the same pipeline with the DCA Poly-encoder included as a final filter for various training data. Comparisons to the Bi-encoder and standard Poly-encoder can be found in Section 6.1, or in detail in Appendix C.

Our system achieves moderate performance on PARSEME and competitive performance on the DiMSUM trained only on the modified SemCor data. When fine-tuned on both MWE identification datasets it further improves, reaching state-of-the-art performance on DiMSUM. Systems fine-tuned on either PARSEME or DiMSUM alone perform even better on their corresponding test set, but worse on the other test set, likely due to differences in domain and MWE type between the datasets.

High precision stands out as a strength of our approach, but it suffers from low recall — even the rule-based pipeline with no model filter lags behind other systems in recall. We attribute this mainly to the issue of lexicon dependence described in Section 8; MWEs missing from our lexicon account for a majority of our false negatives as we show in our error analysis (Section 6.2). These findings echo Savary et al. (2019) on the importance of lexicons for MWE identification, and suggest that there is room to improve performance by expanding the lexicon. While it is difficult to pinpoint exactly why we achieve state-of-the-art F1 on DiMSUM and not PARSEME, one significant difference is that more

[4]The STREUSLE evaluation script rejects input without appropriate lexical categories/supersenses

PARSEME 1.1							DiMSUM		
MWE-based			**Token-based**			**System**	**MWEs**		
P	**R**	**F**	**P**	**R**	**F**		**P**	**R**	**F**
–	–	36.0	–	–	40.2	Taslimipoor+ (2019) Kirilin+ (2016)	73.5	48.4	58.4
–	–	**41.9**	–	–	–	Rohanian+ (2019) Williams (2017)	65.4	**56.0**	60.4
36.1	**45.5**	40.3	40.2	**52.0**	**45.4**	Liu+ (2021)	47.9	52.2	50.0
16.3	39.9	23.1	19.2	43.9	26.7	Rule-based Pipeline	57.7	55.5	56.6
28.2	38.5	32.5±0.4	30.7	39.0	34.3±0.4	Rules + DCA (S)	70.9	53.0	60.6±0.1
35.7	39.3	37.4±0.6	37.7	38.6	38.1±0.4	Rules + DCA (S/D)	78.2	51.8	**62.3**±0.1
47.1	33.8	39.4±0.3	**48.3**	32.1	38.6±0.2	Rules + DCA (S/P)	75.7	49.4	59.8±0.1
45.4	33.2	38.3±0.1	46.9	31.9	38.0±0.2	Rules + DCA ($S/P/D$)	**80.4**	49.5	61.3±0.4

Table 4: Test set results on PARSEME 1.1 English and DiMSUM for MWE identification. All DCA poly-encoder models function as a final filter after the rule-based pipeline. Training data is listed in parenthesis: S=SemCor, P=PARSEME, D=DiMSUM. For trainable models we report the mean (± standard deviation for the F1 score) of three runs with random seeds. Because our system uses gold POS tags/lemmas to look up sense glosses, we compare against systems using gold information where available, such as for Liu et al. (2021) and Kirilin et al. (2016).

than 40% of the DiMSUM test set MWEs are noun phrases, most of which we can detect without relying on a lexicon (as described in Section 4.4.2). For PARSEME, we must always rely on our lexicon.

5.1 WSD Performance

We compare performance on the English WSD all-words task to Blevins and Zettlemoyer (2020), a similar Bi-encoder system trained only for WSD. Recent work in WSD has achieved higher scores (Barba et al., 2021b), but our goal is to understand how the addition of the MWE identification task affects WSD performance.

System	**F1**	**System**	**F1**
Blevins+	79.0	PolyEnc (S)	73.8±0.2
DCA (S)	77.2±0.1	BiEnc (S)	77.4±0.6
DCA ($S/P/D$)	74.4±0.6	BiEnc ($S/P/D$)	74.2±1.0

Table 5: English WSD all-words task F1.

Our system retains most but not all of its WSD performance: F1 is 2% lower when trained on our modified SemCor data and 6% lower when fine-tuned on PARSEME+DiMSUM. We attribute this drop in F1 from fine-tuning to potentially confusing labels in the fine-tuning data: the gold label of positive examples is always the MWE's first sense, which may be incorrect for polysemous MWEs, and as we show in Section 6.2, many negative example MWEs actually have senses appropriate for the context they are in. Consequently, the model cannot rely entirely on matching sense glosses to input contexts for this data and may forget some knowledge useful for WSD.

Comparing models, the DCA Poly-encoder outperforms the standard Poly-encoder on WSD, but its performance does not significantly differ from the Bi-encoder. We leave Poly-encoder architectures better suited for WSD to future work.

6 Analysis

6.1 MWE Identification Ablations

System	**PARSEME**	**Δ**	**DiMSUM**	**Δ**
Rules+DCA	38.3	–	61.3	–
-SemCor Data	26.0	-12.3	56.8	-4.5
-Rule Filters	35.5	-2.8	61.8	+0.5
Rules+BiEnc	36.5	-1.8	61.3	–
Rules+PolyEnc	34.0	-4.3	60.3	-1
Rules	23.1	–	56.6	–
-Filters	14.4	-8.7	47.7[5]	-8.9

Table 7: MWE identification F1 for ablations. Aside from the ablation removing SemCor data, all models are trained on SemCor+PARSEME+DiMSUM.

Pretraining using the modified SemCor data is important; training only on the MWE identification datasets substantially reduces performance. Intuitively, this can be thought of as the model needing to learn how to encode context words and sense glosses before learning to apply that knowledge to MWE identification.

[5]Output for the rule-based pipeline with no filters was invalid according to the DiMSUM grader and had to be approximated, so it may be off by 1-2 F1 points.

Dataset	Type	Sentence	Note
PARSEME	FP	*...were **propped up** on a foot-warmer, ...*	**prop up** never marked as MWE in dataset
PARSEME	FN	***Never mind**, Mrs. Bray will join you later.*	**never mind** missing from lexicon
PARSEME	FP	*...his mind **drifted off** to the accounts...*	**drift off** sense "fall asleep" does not apply
PARSEME	TN	textit...as we **sat** side **by** side...	**sit by** sense "be inactive" does not apply
DiMSUM	FP	*Aww, **thank you**.*	**thank you** marked as MWE in 4 other sentences
DiMSUM	FN	*All our dreams can **come true**,...*	**come true** missing from lexicon
DiMSUM	FN	*...this was a **breathe of fresh air**.*	Present in lexicon; model filter false negative
DiMSUM	TN	*...impact my wardrobe **has on** the environment.*	**have on** sense "dress in" does not apply

Table 6: Representative errors (FP/FN) and incorrect MWEs successfully excluded by the model filter (TN)

We also see that while removing rule-based filters from the DCA pipeline lowers PARSEME F1, it slightly raises DiMSUM F1, suggesting that the necessity of these filters depends on the data. However, removing the rule-based filters only works because the DCA Poly-encoder can accurately exclude false positives: removing the same filters from the purely rule-based pipeline results in a very low F1. Finally, the DCA Poly-encoder substantially outperforms the standard Poly-encoder (PolyEnc) on both datasets and surpasses the Bi-encoder on PARSEME, demonstrating that our DCA Poly-encoder model can improve MWE identification performance.

6.2 Error Analysis

We perform an error analysis on the output of our SemCor trained and fine-tuned models on both test sets, taking 50 false positives and 50 false negatives from each combination of model and dataset (for a total of 400 examples). Select examples can be seen in Table 6, and detailed statistics about the outcome of our analysis can be found in Appendix B.

We find that for >80%[6] of false positives a sense from our lexicon was appropriate for the given context, but the target words were not marked as a MWE in the data. Many of these MWEs were present in our lexicon but nowhere in the test set, suggesting discrepancies between the scope of what WordNet and these datasets respectively define as MWEs. However, there were also a number of false positives that *are* marked as MWEs in other places in the dataset. This could happen if these combinations of words were only marked as MWEs when they had specific meanings or particularly non-compositional semantics, but this was not the case for the examples we examined. These results speak to the difficult and potentially subjective nature of annotating MWEs, and we hope to see work exploring this area in the future.

For false negatives, >85% were cases where the target MWE was missing from the lexicon, confirming that the bottleneck for recall is our system's lexicon. For the majority of the remaining false negatives, an appropriate sense for the given context was present in our lexicon, meaning that these were failures of our MWE identification system and not the lexicon. However, the fact that errors in matching meaning to context account for <20% of false positives and <15% of false negatives shows that our model has successfully learned how to judge whether a group of words constitutes a MWE with a given meaning. See Table 6 true negatives for examples of MWEs excluded based on meaning.

7 Conclusion

In this work, we present an approach to MWE identification using rule-based candidate extraction with a model filter, achieving strong results on the PARSEME 1.1 English data and state-of-the-art results for MWE identification on the DiMSUM dataset. Our system performs both MWE identification and WSD with the same model, demonstrating that these tasks can be tackled together. We also introduce a modified Poly-encoder architecture better suited to MWE identification.

Our system's strength is its high precision for MWE identification. We show its low recall to be a function of lexicon size, and in future work we intend to expand the lexicon by mining MWEs and generating glosses for them, which has the potential to substantially increase recall for lexicon-based systems. Improved approaches for multitask training of MWE identification/WSD models could also be valuable; the ideal pipeline would be competitive with state-of-the-art systems in both tasks, and not just MWE identification.

Ideal applications of our system include MWE identification when a lexicon of target MWEs is available, or cases where quickly performing both

[6]Computed excluding false-positives from the DiMSUM noun phrase detector, which does not use the lexicon

MWE identification and WSD is valuable, such as in language learning and assisted reading tools.

8 Limitations

While our system performs well, the output of our MWE pipeline is limited to MWEs that are present in our lexicon or detectable with simple rules. Furthermore, because our model uses gloss text as input, we cannot effectively filter MWE candidates without sense glosses. Consequently, our approach to MWE identification depends on the presence of a high-quality lexicon which includes MWE lemmas and sense glosses, making it ill-suited for scenarios where data like this may not be available yet, such as in low resource languages. However, we are optimistic that work in MWE discovery (Ramisch et al., 2010) and gloss/definition generation (Bevilacqua et al., 2020) will help to mitigate this problem by automating parts of the data creation process.

9 Acknowledgements

The authors thank the KERNEL organization (DEEPCORE Inc.) for providing the GPUs that made this work possible, Shane Steinert-Threlkeld and Shonosuke Ishiwatari for feedback on early versions of the paper, and Chika Ohe for helping make Figure 1.

References

Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. In *Handbook of Natural Language Processing*.

Edoardo Barba, Tommaso Pasini, and Roberto Navigli. 2021a. ESC: Redesigning WSD with extractive sense comprehension. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4661–4672, Online. Association for Computational Linguistics.

Edoardo Barba, Luigi Procopio, and Roberto Navigli. 2021b. ConSeC: Word sense disambiguation as continuous sense comprehension. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1492–1503, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221, Online. Association for Computational Linguistics.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017, Online. Association for Computational Linguistics.

Morten H. Christiansen and Inbal Arnon. 2017. More than words: The role of multiword sequences in language learning and use. *Topics in cognitive science*, 9 3:542–551.

Mathieu Constant, Gülşen Eryiğit, Johanna Monti, Lonneke van der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. Survey: Multiword expression processing: A Survey. *Computational Linguistics*, 43(4):837–892.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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 Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William Falcon and The PyTorch Lightning team. 2019. PyTorch Lightning.

Vasiliki Foufi, Luka Nerima, and Éric Wehrli. 2017. Parsing and MWE detection: Fips at the PARSEME shared task. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 54–59, Valencia, Spain. Association for Computational Linguistics.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*.

Kamil Kanclerz and Maciej Piasecki. 2022. Deep neural representations for multiword expressions detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 444–453, Dublin, Ireland. Association for Computational Linguistics.

Minju Kim, Chaehyeong Kim, Yong Ho Song, Seungwon Hwang, and Jinyoung Yeo. 2022. BotsTalk: Machine-sourced framework for automatic curation of large-scale multi-skill dialogue datasets. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5149–5170, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Angelika Kirilin, Felix Krauss, and Yannick Versley. 2016. ICL-HD at SemEval-2016 task 10: Improving the detection of minimal semantic units and their meanings with an ontology and word embeddings. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 937–945, San Diego, California. Association for Computational Linguistics.

Harsh Kohli. 2021. Training bi-encoders for word sense disambiguation. *CoRR*, abs/2105.10146.

Nidhi Kulkarni and Mark Finlayson. 2011. jMWE: A Java toolkit for detecting multi-word expressions. In *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World*, pages 122–124, Portland, Oregon, USA. Association for Computational Linguistics.

Murathan Kurfalı and Robert Östling. 2020. Disambiguation of potentially idiomatic expressions with contextual embeddings. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 85–94, online. Association for Computational Linguistics.

Zichao Li, Prakhar Sharma, Xing Han Lu, Jackie Cheung, and Siva Reddy. 2022. Using interactive feedback to improve the accuracy and explainability of question answering systems post-deployment. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 926–937, Dublin, Ireland. Association for Computational Linguistics.

Nelson F. Liu, Daniel Hershcovich, Michael Kranzlein, and Nathan Schneider. 2021. Lexical semantic recognition. In *Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021)*, pages 49–56, Online. Association for Computational Linguistics.

Marco Maru, Simone Conia, Michele Bevilacqua, and Roberto Navigli. 2022. Nibbling at the hard core of Word Sense Disambiguation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4724–4737, Dublin, Ireland. Association for Computational Linguistics.

Marek Maziarz, Stan Szpakowicz, and Maciej Piasecki. 2015. A procedural definition of multi-word lexical units. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 427–435, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(1):39–41.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Ines Montani and Matthew Honnibal. Prodigy: A modern and scriptable annotation tool for creating training data for machine learning models.

Andrea Moro and Roberto Navigli. 2015. SemEval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, Colorado. Association for Computational Linguistics.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231, Atlanta, Georgia, USA. Association for Computational Linguistics.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24, Toulouse, France. Association for Computational Linguistics.

Caroline Pasquer, Agata Savary, Carlos Ramisch, and Jean-Yves Antoine. 2020. Verbal multiword expression identification: Do we need a sledgehammer to crack a nut? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3333–3345, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic. Association for Computational Linguistics.

Manuel F. Pulido. 2022. Why are multiword units hard to acquire for late l2 learners? insights from cognitive science on adult learning, processing, and retrieval. *Linguistics Vanguard*, 8:237 – 247.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110, Valencia, Spain. Association for Computational Linguistics.

Carlos Ramisch, Silvio Ricardo Cordeiro, Agata Savary, Veronika Vincze, Verginica Barbu Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, Tunga Güngör, Abdelati Hawwari, Uxoa Iñurrieta, Jolanta Kovalevskaitė, Simon Krek, Timm Lichte, Chaya Liebeskind, Johanna Monti, Carla Parra Escartín, Behrang QasemiZadeh, Renata

Ramisch, Nathan Schneider, Ivelina Stoyanova, Ashwini Vaidya, and Abigail Walsh. 2018. Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 222–240, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Carlos Ramisch, Aline Villavicencio, and Christian Boitet. 2010. mwetoolkit: a framework for multiword expression identification. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Omid Rohanian, Shiva Taslimipoor, Samaneh Kouchaki, Le An Ha, and Ruslan Mitkov. 2019. Bridging the gap: Attending to discontinuity in identification of multiword expressions. 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 2692–2698, Minneapolis, Minnesota. Association for Computational Linguistics.

Ivan Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for nlp. pages 1–15.

Agata Savary, Silvio Cordeiro, and Carlos Ramisch. 2019. Without lexicons, multiword expression identification will never fly: A position statement. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, pages 79–91, Florence, Italy. Association for Computational Linguistics.

Nathan Schneider, Dirk Hovy, Anders Johannsen, and Marine Carpuat. 2016. SemEval-2016 task 10: Detecting minimal semantic units and their meanings (DiMSUM). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 546–559, San Diego, California. Association for Computational Linguistics.

Nathan Schneider, Jena D. Hwang, Vivek Srikumar, Jakob Prange, Austin Blodgett, Sarah R. Moeller, Aviram Stern, Adi Bitan, and Omri Abend. 2018. Comprehensive supersense disambiguation of English prepositions and possessives. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 185–196, Melbourne, Australia. Association for Computational Linguistics.

Benjamin Snyder and Martha Palmer. 2004. The English all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain. Association for Computational Linguistics.

Yang Song, Xin Cai Ong, Hwee Tou Ng, and Qian Lin. 2021. Improved word sense disambiguation with enhanced sense representations. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4311–4320, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shiva Taslimipoor, Omid Rohanian, and Le An Ha. 2019. Cross-lingual transfer learning and multitask learning for capturing multiword expressions. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, pages 155–161, Florence, Italy. Association for Computational Linguistics.

Jake Williams. 2017. Boundary-based MWE segmentation with text partitioning. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 1–10, Copenhagen, Denmark. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Andrea Zaninello and Alexandra Birch. 2020. Multiword expression aware neural machine translation. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3816–3825, Marseille, France. European Language Resources Association.

Guobiao Zhang, Wenpeng Lu, Xueping Peng, Shoujin Wang, Baoshuo Kan, and Rui Yu. 2022. Word sense disambiguation with knowledge-enhanced and local self-attention-based extractive sense comprehension. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4061–4070, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

A Implementation Details

Bi-encoder and Poly-encoder models are implemented and trained with Pytorch Lightning (Falcon and The PyTorch Lightning team, 2019), using pretrained BERT models from the Transformers library (Wolf et al., 2020). In particular, we use *bert-base-uncased* as the base model for both encoders. We define batch size by the number of training examples (words or MWEs to be labeled) in each batch, and keep this number constant by adjusting the number of sentences and/or masking out examples to save them for the next batch. Our effective batch size is 32. All models were trained on

a single GeForce GTX TITAN X GPU, with hyperparameters tuned using Weights & Biases (Biewald, 2020) to run random sweeps and track performance. Separate sweeps were run for the Bi-encider and Poly-encoder, each having a maximum of 20 runs and using early stopping to terminate runs with poor performance. Our total compute time was approximately 160 days, though this would have been significantly lower using a newer model of GPU. Our models have 220M parameters, and fully training for 15 epochs on the modified SemCor data takes approximately 1.2×10^{17} FLOPS. We used Prodigy (Montani and Honnibal) as our annotation tool. Further detail, including all training hyperparameters and instructions for reproduction, can be found in our published code.

B Error Analysis Details

This appendix contain details about the frequency with which we found various types of false positives or false negatives in our error analysis.

B.1 PARSEME

In the table below, **Def?** represents the % of false positives where a sense appropriate for the predicted MWE was present in our lexicon. **MWE?** represents the % of false positives where the MWE was present in other sentences in the dataset, and the % of false negatives where it was present in our lexicon, respectively.

	False Positives		False Negatives
Model	**Def?**	**MWE?**	**MWE?**
SemCor	90%	16%	6%
fine-tuned	90%	34%	16%

Table 8: PARSEME Error Analysis

B.2 DiMSUM

Our results on DiMSUM are similar to those of PARSEME, except that for the system using the SemCor model 22% of the false positives were from the rule-based consecutive noun tagger, with that number increasing to 56% for the fine-tuned model (the false positive rate drops substantially after fine-tuning the filtering model as can be seen in Table 4, which leads to these errors accounting for a higher percentage of total false positives). The **Def?** and **MWE?** percentages for false positives in the below table are computed excluding consecutive noun tagger false positives.

	False Positives		False Negatives
Model	**Def?**	**MWE?**	**MWE?**
SemCor	92%	56%	4%
fine-tuned	81%	63%	12%

Table 9: DiMSUM Error Analysis

C Detailed Performance

Table 10 below contains full scores for systems omitted from the main paper for brevity.

System	PARSEME 1.1						DiMSUM			WSD
	MWE-Based			**Token-based**			**MWEs**			
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**F1**
Rules (no filters)	8.8	38.9	14.4	12.0	49.9	19.3	40.5*	58.0*	47.7*	–
Rules (both filters)	16.3	39.9	23.1	19.2	43.9	26.7	57.7	55.5	56.6	-
BiEnc (S)	27.5	38.8	32.2±0.8	30.0	39.43	34.1±0.3	70.7	52.57	60.0±0.4	77.4±0.6
BiEnc ($S/P/D$)	44.5	31.0	36.5±0.9	46.4	30.5	36.2±0.8	80.9	49.3	61.3±0.4	74.2±1.0
PolyEnc (S)	27.1	36.1	30.9±0.3	29.8	37.1	33.0±0.2	69.7	51.7	59.3±0.2	73.8±0.2
PolyEnc ($S/P/D$)	37.7	31.0	34.0 ±0.7	40.7	31.2	35.3±0.4	78.0	49.1	60.3±0.2	66.0±0.2
DCA (S)	28.2	38.5	32.5±0.4	30.7	39.0	34.3±0.4	70.9	53.0	60.6±0.1	77.2±0.1
DCA (S/D)	35.7	39.3	37.4±0.6	37.7	38.6	38.1±0.4	78.2	51.8	62.3±0.1	75.6±0.1
DCA (S/P)	47.1	33.8	39.4±0.3	48.3	32.1	38.6±0.2	75.7	49.4	59.8±0.1	76.4±0.1
DCA ($S/P/D$)	45.4	33.2	38.3±0.1	46.9	31.9	38.0±0.2	80.4	49.5	61.3±0.4	74.4±0.6
DCA ($_{S/P/D, \text{no filters}}$)	40.2	31.8	35.5±0.9	42.7	31,4	26.2±0.5	80.0	50.4	61.8±0.5	74.4±0.6
DCA (P/D)	25.4	26.8	26.0	28.5	27.9	28.2	74.7	45.7	56.8	39.5

Table 10: Test set results on PARSEME 1.1 English and DiMSUM for MWE identification, and the English all-words WSD task. For MWE identification, all Bi-encoder (BiEnc) and and Poly-encoders (PolyEnc/DCA) function as a final filter in the rule-based pipeline. Letters after system entries indicate training data, where S = SemCor, P = PARSEME and D = DiMSUM. For example, ($S/P/D$) means trained on SemCor and finetuned on PARSEME and DiMSUM. Scores marked with the asterisk * come from pipeline configurations that did not produce valid output according to the DiMSUM scorer and had to be approximated, so they may be off by 1–2 F1 points.