Killing Four Birds with Two Stones: Multi-Task Learning for Non-Literal Language Detection

Erik-Lân Do Dinh, Steffen Eger, and Iryna Gurevych
Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
http://www.ukp.tu-darmstadt.de

Abstract

Non-literal language phenomena such as idioms or metaphors are commonly studied in isolation from each other in NLP. However, often similar definitions and features are being used for different phenomena, challenging the distinction. Instead, we propose to view the detection problem as a generalized non-literal language classification problem. In this paper, we investigate multi-task learning for related non-literal language phenomena. We show that in contrast to simply joining the data of multiple tasks, multi-task learning consistently improves upon four metaphor and idiom detection tasks in two languages, English and German. Comparing two state-of-the-art multi-task learning architectures, we also investigate when soft parameter sharing and learned information flow can be beneficial for our related tasks. We make our adapted code publicly available[1].

1 Introduction

As Lakoff and Johnson (1980) argued, metaphorical concept mappings, often from concrete to more abstract concepts, are ubiquitous in everyday life, thus they are ubiquitous in written texts. But the same is true for other kinds of so-called *non-literal language*, e.g., for idioms. Boundaries between these instances of non-literality are not always clear: for instance, "[...] a swathe of sunlight lay across the red-tiled floor."[2] can be classified as a metaphor because of its non-lexicalized figurative meaning. Few would dispute the idiomaticity of "kicking the bucket", as the non-literal meaning in this multi-word expression is largely conventionalized. However, in the sentence "One approach would be to draw the line by reference [...]" the expression "draw the line" could be classified as either metaphorical (because it still evokes the literal senses of its constituents) or idiomatic (as it is a fixed expression with lexicalized figurative sense).

Much effort has been spent on the detection of metaphors, idioms, and general non-literal language use (Shutova, 2015). However, because of the named vague and subjective nature of these phenomena, a multitude of datasets using differing definitions has emerged in the process. Even when datasets address the same aspect of non-literality, they may use diverging or underspecified definitions; compare, e.g., the guidelines for annotating metaphors of Tsvetkov et al. (2014) ("[...] all words that, in your opinion, are used non-literally [...]") and Mohammad et al. (2016) ("more complex; more distant from our senses; more abstract; more vague; ...").

The fuzziness of non-literality has two natural consequences: (1) training data is sparse, because different researchers may use diverging definitions, and hence may annotate different things rather than extend "the same story"; (2) high-quality training data is costly to obtain because there may be considerable disagreement among crowd-workers and even trained experts regarding the labels for different instances of (non-)literality.

In this work, we address two research questions:

[1] https://github.com/UKPLab/coling2018-nonliteral-mtl

[2] VUAMC (Steen et al., 2010)

Proceedings of the 27th International Conference on Computational Linguistics, pages 1558–1569
Santa Fe, New Mexico, USA, August 20-26, 2018.

- (how) can we leverage existing datasets for non-literality, possibly using diverging definitions and addressing different aspects of non-literality, to bootstrap classifiers on new sparse data? This question is particularly relevant for non-literal language detection due to problem (2) above;

- do existing datasets for non-literality share common ground in the first place or are many of them using arbitrary and mutually exclusive definitions? If the answer to this question is "yes", then this would call for a re-thinking of a whole computational research area, and a return to safe grounds.

To address our research questions, we apply *multi-task learning* (MTL) to four non-literal language datasets. Introduced by Caruana (1993), MTL has recently been used to improve a variety of NLP tasks with the help of other, auxiliary tasks. This includes chunking (Søgaard and Goldberg, 2016), using POS tagging, and spelling normalization (Bollmann and Søgaard, 2016), using orthographic data from other domains, but also semantic tasks like entailment (Hashimoto et al., 2017) or argument mining (Schulz et al., 2018). The underlying idea of MTL is that it is beneficial to learn several tasks jointly because of "spill-over" effects. Some researchers have claimed that a requirement for success in MTL is task relatedness. If this is true, then MTL is a formidable testbed for *both* of our research questions.

We aim at **four birds**, namely: We consider detection of non-literal phenomena in four different datasets, regarding each as a separate task. These are (A) *metaphor detection in content tokens*, (B) *classification of metaphorical adjective-noun constructions*, (C) *detection of idiomatic use of infinitive-verb compounds*, and (D) *non-literal usage of particle verbs*. Two of the datasets comprise English data, the other two consist of German data.

We throw **two stones** at them, namely: (i) an MTL sequence tagging framework using hard-parameter sharing (Kahse, 2017) and (ii) Sluice Networks (Ruder et al., 2017), for which sharing of information is not hard-wired, but can adjust softly. Both frameworks yield different insights because the first has to make use of all available data, while the second can freely decide if other tasks contain relevant information and if so, how much sharing to enable.

This work is structured as follows. We first ground our motivation to tackle four non-literal language tasks in Section 2. There, we also discuss applications of MTL, from pure syntactic to higher-level, semantic tasks. In Section 3, we describe the two systems that we compare in our experiments. We specify the used datasets and describe their differences in Section 4. In Section 5, we discuss our experiments and results. Finally, we conclude with an outlook on possible future work in Section 6.

2 Related Work

Work in non-literal or figurative language classification usually revolves around one specific phenomenon, be it metaphor, idiom or also metonymy detection. While many approaches are monolingual, some explore cross-lingual non-literal language classification. Tsvetkov et al. (2014) train models separately on English adjective-noun and subject-verb-object metaphors using random forests and a variety of semantic features, including supersenses and concreteness values. The models are then used to classify metaphors in similarly annotated Spanish, Russian, and Farsi test sets.

To the best of our knowledge, a combined detection of multiple non-literal phenomena has not been conducted before. This is surprising because common semantic features have already been used to classify different kinds of non-literal language.

One such typical feature in non-literal language classification is the violation of selectional preferences (Wilks, 1978). It is used, e.g., in *met** (Fass, 1991) to classify metaphors and metonyms. While the system distinguishes between both phenomena, it does so only after using the selectional preference information. In a related task, Horbach et al. (2016) employ this information for classifying idiomatic uses of infinitive verb compounds. Another feature used across different non-literal language detection tasks is topic information. While the work by Horbach et al. (2016) includes this feature for idiom detection, Beigman Klebanov et al. (2014) utilize it to classify metaphorically used tokens. Additionally, they make use of concreteness ratings, grounded in the Conceptual Metaphor Theory (Lakoff and Johnson, 1980). However, as argued in our introduction, concreteness is also useful for the detection of other kinds of non-literal language. For example, Zhang and Gelernter (2015) utilize such ratings to detect

metonymy. Further, supersenses are employed to detect metaphors (Tsvetkov et al., 2014) or non-literal language in general (Köper and Schulte im Walde, 2017). One more feature that is often integrated is textual cohesion, e.g., in metaphor (Schulder and Hovy, 2014) and idiom detection (Sporleder and Li, 2009). The use of such common features suggests that different aspects of non-literality require similar information and that representation sharing may thus turn out beneficial.

Introduced in the early nineties (Caruana, 1993), multi-task learning (MTL) has been more widely and successfully used in NLP recently (Ruder, 2017). MTL denotes a machine learning technique in which multiple tasks are trained in parallel in the same system, using a shared representation. The goal is to take advantage of commonalities between the different tasks. Bollmann and Søgaard (2016) use multi-task learning for historical spelling normalization. They employ texts from different domains as main respectively auxiliary tasks and improve upon a CRF baseline. Søgaard and Goldberg (2016) show that task supervision at lower layers for lower-level syntactic tasks like POS-tagging is beneficial to higher-level syntactic tasks such as chunking and CCG supertagging. Since their experiments with semantic tasks (NER, supersense tagging) on the higher levels show no improvement, they conclude that tasks need to be "sufficiently similar, e.g., all of syntactic nature" for multi-task learning to increase performance. Their conclusion is challenged by Hashimoto et al. (2017), who create a similar multi-task learning network in which lower layers predict syntactic tasks, while higher layers predict sentential relatedness and entailment. Their semantic tasks also improve when introducing *shortcut connections*, i.e., feeding the word representations into all layers of their network. In contrast, Alonso and Plank (2017) find generally mixed performance of MTL for semantic tasks. They also use syntactic tasks as low-level auxiliary tasks, but cannot improve performance over a single-task learning baseline for three out of five investigated semantic tasks (NER, supersense classification, frame identification). Schulz et al. (2018) apply MTL to another semantic task, argumentation mining. Instead of syntactic tasks as auxiliaries, they use diverse argumentation mining datasets from different domains. Similar to our tasks, annotation and labels vary across their different tasks due to inherent subjectivity and vagueness of argumentation mining (as for non-literal language detection). Experimenting with artificially downsized datasets, they show that—especially when data for the main task is sparse—MTL improves sequence tagging results for argumentation mining, even when the tasks cover different domains. In contrast, we also investigate the effect of auxiliary tasks with small datasets on a large-data main task.

3 System Descriptions

We conduct our multi-task learning experiments using two different architectures: a sequence tagging framework (MTL-SEQ) (Kahse, 2017), and Sluice Networks (SLUICE) (Ruder et al., 2017).

We adapt both systems to our datasets, which only have labels for few tokens in a sentence. Thus, overfitting on the missing/"empty" labels seems probable, and is confirmed by preliminary experiments. To alleviate this issue, we exclude the loss on the tokens with empty labels from the total loss calculation, so that they do not influence weight updates.

Multi-Task Learning Sequence Tagging Framework (MTL-SEQ) We employ the system by Kahse (2017), a framework for multi-task learning sequence tagging generalizing the model of Søgaard and Goldberg (2016). An example of a two-task setup is shown in Figure 1. It uses English metaphor classification on a token level as the main task, and English metaphor detection in adjective-noun constructions as an auxiliary task (both described in Section 4).

After an input layer that reads in word embeddings, there are multiple shared, bi-directional LSTM layers, thus implementing hard parameter sharing. The shared layers are followed by a number of fully connected task-specific layers (Peng et al., 2017), storing private information for each task. At their end, a softmax classifier predicts labels for each input token.

In addition to using pre-trained word embeddings, the architecture incorporates character-level information to improve handling of out-of-vocabulary words. The framework can further be configured to terminate different tasks at different layers. We set this option to randomly use one of two scenarios: either all tasks use all BiLSTM layers, or all auxiliary tasks terminate one layer before the main task.

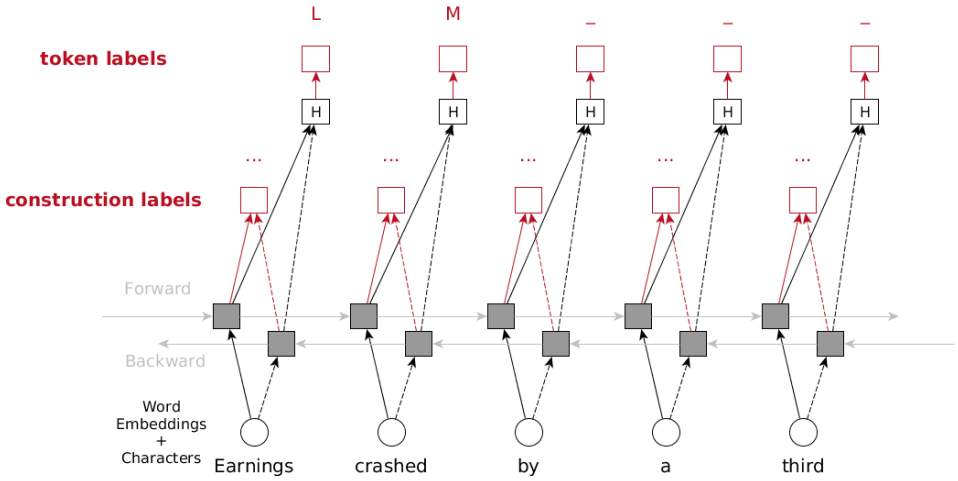

Figure 1: Example setup for the sequence tagging MTL framework (Kahse, 2017). It shows auxiliary task `adj-noun-met` and main task `tok-met`. Gray blocks represent shared BiLSTM units. In this example, both tasks are terminated after one shared BiLSTM and multiple private fully connected layers (white *H* blocks). The input is task-specific and only evaluated for the respective task; pictured is input for `tok-met`. "Earnings" is labeled as literal, "crashed" is labeled as a metaphor; the remaining tokens have no label and their loss is excluded from the total loss calculation.

We optimize over several hyper-parameters, which include the number of task-specific fully connected layers, the layer at which tasks should terminate, and word dropout rate. An overview of the hyper-parameters and the range from which we randomly sample their values is shown in Table 1.

Sluice Networks (SLUICE) Ruder et al. (2017) introduced *sluice networks* as a generalization and combination of various MTL architectures that use different information transfer techniques. Thus, it can emulate, e.g., hard parameter sharing (Caruana, 1993), where the same layer weights (i.e., the same layers) are used for different tasks, and cross-stitch networks (Alonso and Plank, 2017), in which trainable parameters decide the amount of shared information between layers of separate networks. A generic example is given in Figure 2.

The system first reads input in an embedding layer, and combines the word embeddings with character-level embeddings (produced by a BiLSTM). This representation is passed to task-specific BiLSTM networks, which learn to share parts of their layer outputs, i.e., activations, in the following way. Each layer contains two different subspaces, to store both task-specific, and shared representations. Trainable parameters α determine the amount of sharing between the subspaces of the task BiLSTMs. Parameters β decide to which degree the output of the BiLSTM layers should be used for the task-specific classifier (a multilayer perceptron). This way, they allow for hierarchical relations.

Most hyper-parameters regarding information and weight sharing are encoded into trainable parameters for sluice networks. Thus, the most important hyper-parameters are the number of BiLSTM layers in the network, the way layer output is combined, and constraints on the subspaces (see Table 2).

Hyper-parameter	value range
number of of shared BiLSTM layers	{1, 2, 3}
number of task specific fully connected layers	{0, 2}
Word dropout	[0.0, 0.5]
Dropout	[0.0, 0.5]
Activation function for fully connected layers	{tanh, relu}
Output layer	{0, 1, 2}

Table 1: Hyper-parameters for the multi-task learning sequence tagging framework (Kahse, 2017).

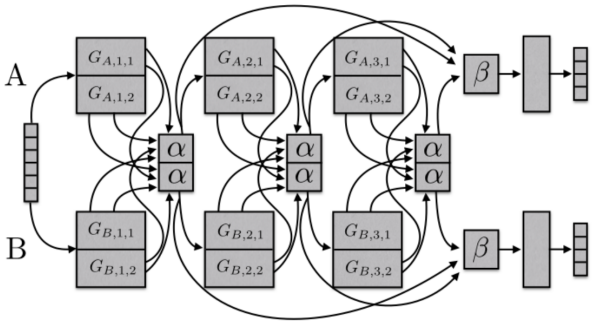

Figure 2: Example for sluice networks (Ruder et al., 2017). A single input representation is passed into task networks A and B. Both of the two networks i have three RNN layers j, consisting of two subspaces $G_{i,j,k}$ each. For each layer output, α determines the amount of sharing between the layers of different tasks while β controls the layer at which a task network terminates.

Conceptually, the largest difference between both systems is the following: MTL-SEQ implements hard parameter sharing, i.e., the layer weights are the same for all tasks, and the structure of the network is predefined; in particular, at which layer a task should terminate. Essentially, this amounts to using one network for multiple tasks, trained jointly. In contrast, SLUICE has separate layers for each task, and learns parameters which control the information flow between those task-specific layers. Further, the level at which task predictions take place is also learned for each task. This means that two networks are trained in parallel, that share a learnable amount of information. Comparing both approaches, we can analyze which tasks profit from which degree of information sharing.

Hyper-parameter	value range	notes
number of BiLSTM layers	{2, 3}	
layer connections	{stitch, skip}	*stitch* denotes learnable β parameters, while *skip* only uses the last BiLSTM layer as classification input
subspace constraint weights	{0.1, 0.2}	

Table 2: Hyper-parameters for the sluice networks.

4 Datasets and Tasks

In our experiments, we consider four different non-literal language tasks (Table 3). Specifically, we include two metaphor detection tasks, with data being annotated on a token and on a construction level (i.e., grammatical constructions that share one label). We further include data from two tasks classifying idiomatic language use. The latter two tasks use German data, while the metaphor task datasets are in English. While those tasks all cover distinct, or at least distinctly annotated, non-literal language phenomena, on a textual level they all are examples of irregular polysemy. Thus, in many feature-based detection systems similar or even the same features and resources are used to classify these phenomena.

Token-level Metaphor Classification We consider metaphor detection on a token level, i.e., each (content) token should be classified as belonging to a metaphor or not. We employ the *VU Amsterdam Metaphor Corpus* (Steen et al., 2010), short *VUAMC*, for our experiments. It is the largest available resource in which each token is specifically annotated as being used metaphorically or literally, and comprises four genres: *academic*, *conversation*, *fiction*, and *news* texts.

The VUAMC is annotated using very rigid annotation guidelines, *MIPVU* (Steen et al., 2010). In short, each token that is not used in its most basic (i.e., most concrete, body-related, historically oldest) sense is labeled as metaphoric. Unclear cases are resolved using specified online dictionaries. These guidelines lead to a high inter-annotator agreement of 0.84 Fleiss' κ. At the same time, they introduce the problem of very commonly used word senses being annotated as metaphoric. For example, prepositions such

Task	dataset	size	M	lang	example
Token-level metaphor detection	VUAMC	103,865	15%	en	"Along with Sir James he **found** the US much more **attractive**, [...]"
Construction level metaphor detection	Tsvetkov et al., (2014)	1,738	47%	en	Wind and wave power providing the **green energy** of the future.
Classification of idiomatically used verb compounds	Horbach et al., (2016)	5,249	64%	de	"Auch eine Uhr, die **stehen geblieben** ist, geht zweimal am Tag richtig", sagt er. ("A clock that has **stopped running**, is correct two times a day, too," he says.)
Classification of non-literally used particle verbs	Köper and Schulte im Walde (2016)	6,436	35%	de	Auf Decken sitzt man ums Feuer und lässt den ereignisreichen Tag **nachklingen**. (You sit around the fire on blankets and let the day **linger on** [lit.: ring on].)

Table 3: Investigated tasks and datasets. *Size* describes labeled tokens in case of token-level metaphor detection (content tokens), and labeled constructions for the other tasks respectively, *M* denotes percentage of non-literal labels. Non-literal use of tokens/constructions in the examples is marked bold.

as *on* are labeled as metaphoric when not being used in their positional sense. We partially avoid this problem by filtering out non-content words and auxiliary verbs. Dataset statistics are given in Table 3.

Construction-level Metaphor Classification Many datasets in metaphor detection tasks label *grammatical constructions* instead of tokens, prominently, e.g., adjective-noun compounds or subject-verb/verb-object compounds. We use the adjective-noun dataset by Tsvetkov et al. (2014). Their training set was created by two annotators and later curated, while the test set—comprised of sentences from the TenTen web corpus (Jakubíček et al., 2013)—was annotated by five annotators. Fleiss' κ for the latter is reported as 0.74.

Compared to the VUAMC, the guidelines for this dataset are substantially less specific, appealing to the intuition of the annotators (in fact, annotators were asked to mark all *non-literally* used words). Due to this difference in guidelines and the focus on adjective-noun metaphors, the resulting annotations also differ from the VUAMC, which is why we consider this a different task.

For the test set, the original 200 sentences containing the construction are available. We automatically crawl large corpora[3] to obtain sentences for the constructions in the training set. Note that both training and test sets are substantially smaller than those of the VUAMC (Table 3), by a factor of 50.

Classification of idiomatically used infinitive-verb compounds In contrast to the metaphor datasets which can also include more novel meanings of the annotated words, e.g., "spot of information", Horbach et al. (2016) focus on idiomatic usage of verb compounds. This means they consider only conventionalized figurative meanings, i.e., compounds whose figurative sense is already lexicalized. They create a corpus of literal and idiomatic uses of six German infinitive-verb compounds (e.g., *sitzen lassen—to leave sitting/abandon*). Thus, this dataset contains considerably more samples for a specific compound (up to 1,000 for each) compared to the previous two datasets. Genres covered are newspaper and magazine articles published from 1993–2013.

Two lexicographers were tasked to annotate occurrences of the compounds within a three-sentence

[3]ukWac (Baroni et al., 2009), British National Corpus (BNC Consortium, 2007)

context as *literal* or *idiomatic*, but were given no further guidelines. Still, they achieve an inter-annotator agreement of 0.72 Cohen's κ. The dataset contains the unanimously labeled instances and a portion of adjudicated ambiguous cases.

Classification of non-literally used particle verbs Similar to the infinitive-verb compound dataset, Köper and Schulte im Walde (2016) investigate pre-set constructions, specifically 165 different German particle verbs. For each particle verb, they selected up to 50 sentences containing *literal* and *non-literal* uses from a German web corpus, *DECOW14AX* (Schäfer and Bildhauer, 2012; Schäfer, 2015); however, they do not discuss the types of non-literality covered. Annotation was conducted by three German speakers with "linguistic background" on a 6-point scale; guidelines are not disclosed. Köper and Schulte im Walde (2016) report an inter-annotator agreement of 0.70 Fleiss' κ after mapping the 6-point scale to a binary annotation.

5 Experiments

We conduct three types of experiments for MTL-SEQ and SLUICE:

- a regular single-task learning baseline *(STL)*
- a combined-data single-task learning baseline *(MERGED)*, where we merge the training data of the different tasks
- a multi-task learning setup *(MTL-all)*, where one main task is supported by three auxiliary tasks

The MERGED baseline allows us to differentiate between causes for possible improvements, since we want to investigate in which way additional tasks and data can influence performance. We further partition our MTL experiments along languages, i.e., we carry out additional experiments separately for German *(MTL-de)* and English *(MTL-en)* tasks. Those mono-lingual setups thus only include one auxiliary task.

Setup We use the same train and test split as in Tsvetkov et al. (2014) for `adj-noun-met`. Thus, we use 11% of the data as test data and the remainder as training data. We use 20% of the training data as development data. For `tok-met`, we use a split of 70%/20%/10%. For the two German idiom datasets, we use a training/development/test set split of roughly 80%/10%/10%.

As input to both neural networks we use 100-dimensional, bilingual word embeddings trained using bivec (Luong et al., 2015) on a parallel (en-de) version of the Europarl corpus (Koehn, 2005).

Results We report test set results for the best performing system configurations on the development sets and use macro F_1-score to compare our results. We first analyze the results of the different architectures separately, before comparing both approaches.

Task type	`tok-met`	`adj-noun-met`	`inf-verb`	`part-verb`
STL	0.4399	0.5172	0.8507	0.7037
MERGED	0.4600	0.5976	0.8405	0.6875
MTL-all	**0.4897**	**0.6705**	**0.8602**	**0.7083**
MTL-en	0.4277	0.5357	–	–
MTL-de	–	–	0.8519	0.7065

Table 4: **MTL-SEQ** F_1-scores on the test sets for the different tasks and experiment types. *MTL-all* specifies that the respectively remaining three tasks are used as auxiliary tasks. *MTL-en* and *MTL-de* stand for mono-lingual experiments, i.e., containing only one auxiliary task.

Task type	tok-met	adj-noun-met	inf-verb	part-verb
STL	0.4833	0.5487	0.8609	0.7894
MERGED	0.4570	0.5143	0.8401	0.7965
MTL (all)	**0.5604**	**0.6333**	0.8763	0.8007
MTL (en)	0.5467	0.6207	–	–
MTL (de)	–	–	**0.8833**	**0.8146**

Table 5: **SLUICE** F_1-scores on the test sets for all four tasks and experiment types.

MTL-SEQ Our results for MTL-SEQ are given in Table 4.

For tok-met, MERGED already improves over the STL baseline. MTL-all further improves upon the two. In contrast, including only adj-noun-met as auxiliary task (MTL-en) performs even worse than STL.

For adj-noun-met, the pattern is identical except that MTL-en slightly outperforms STL. Comparing MERGED with MTL-all we see that the performance difference is mainly due to MTL-all introducing fewer new classification errors over the STL baseline.

In other words: both metaphor tasks, where training (and evaluation) data for a particular token or construction is sparse, profit heavily from inclusion of the idiom datasets. In contrast, using only the respective other English metaphor dataset does not help. Here, the difference between STL and MTL-en is not statistically significant (McNemar's Test, p = 0.22). It is notable that the German idiom datasets improve the English metaphor tasks by jointly fitting the same network.

The results for inf-verb and part-verb display a different pattern. MERGED performs worse than STL and even though MTL-all improves over both baselines, the difference is not significant. Unlike for the English datasets, MTL-de performs better than STL. The network can better fit to the datasets which are designed to contain many instances of a particular token or construction (inf-verb and part-verb). This is evidenced by the large differences in F_1-scores between the tasks. But these idiom datasets also substantially improve classification for the English metaphors, regardless of differences in language and annotated phenomenon.

SLUICE Results for SLUICE are found in Table 5.

For tok-met, the MERGED baseline performs weaker than STL. In contrast, both MTL approaches improve substantially over the baselines, by more than 6 percentage points over STL. Further, MTL-all outperforms MTL-en, although not significantly so.

Similar to tok-met, for adj-noun-met MTL-all improves significantly over STL. The performance drop from STL to MERGED is similarly large. Also, MTL-all outperforms MTL-en here, even though not significantly. Even though the difference in F_1-score between MTL-all and MTL-en is small, they make considerably different predictions. 16 instances were labeled correctly using MTL-all, but misclassified by MTL-en; and the same number vice versa. Thus, 16% of all 200 test instances were classified differently by the MTL setups. This means that adding the additional idiom tasks does indeed add further information that is helpful for some instances, but detrimental to others—more so than for tok-met.

For inf-verb, STL again outperforms MERGED. MTL-all and MTL-de marginally improve upon the baselines, with the latter performing best, though slightly. Again, both MTL setups show differences in the correctly labeled instances, albeit less than it was the case for adj-noun-met (12%). These differences are distributed similarly over the six compounds. An exception is *sitzen bleiben (to stay seated/to be stuck with something)*, which is considerably better classified using MTL-de than MTL-all.

Results for part-verb differ from the previous tasks, as the MERGED baseline actually perform better than STL. Still, MTL-all and MTL-de improve upon both baselines, with the mono-lingual approach again performing slightly better.

STL		actual	
		M	L
predicted	M	1,396	975
	L	2,580	20,660

MTL-all		actual	
		M	L
predicted	M	1,622	1,027
	L	2,354	20,608

Table 6: `tok-met`: Confusion matrices for MTL-SEQ.

STL		actual	
		M	L
predicted	M	1,567	942
	L	2,409	20,745

MTL-all		actual	
		M	L
predicted	M	2,119	1,467
	L	1,857	20,220

Table 7: `tok-met`: Confusion matrices for SLUICE.

Analysis For both MTL-SEQ and SLUICE, the MTL-all approach outperforms both STL and MERGED baselines. Contrary to MTL-SEQ, SLUICE performs worse on the MERGED baseline than STL for the English tasks.

Consistent between both architectures is that they show significant—and indeed, very substantial—performance gains using MTL for the metaphor datasets. For the idiom and non-literal language datasets, while also improving using MTL, increases are smaller. We attribute this to the nature of the datasets `inf-verb` and `part-verb`, as both comprise multiple annotated instances of few particular compounds/verbs. Thus, the STL approach already yields good performance, and adding the more unstructured information of the metaphor datasets does not improve the MTL settings.

An important difference can be observed in the specific MTL setups. While MTL-SEQ fails to profit from just one (in-language) auxiliary task, SLUICE improves significantly over STL even in this lower-resource setup. Adding more auxiliary tasks boosts MTL-SEQ, but does not change SLUICE performance significantly. For MTL-SEQ, this indicates that including information from just one auxiliary task skews the main task too much in one direction, due to hard parameter sharing, while including multiple auxiliary tasks appears to better "balance" the architecture. In contrast, SLUICE can choose to only share the information necessary to improve the tasks, and thus already profits heavily from one auxiliary task. While we expected an increase in performance for the small-data task `adj-noun-met` using more data-heavy auxiliary tasks, it is interesting that `tok-met` also yields better results when including `adj-noun-met` as auxiliary. We attribute this to the network successfully learning a common representation.

We investigate the improvements of MTL-all over STL on the `tok-met` task, using the confusion matrices of MTL-SEQ (Table 6) and SLUICE (Table 7). For both networks, the increase in F_1-score mainly arises due to an increase in recall. This advantage, especially the increase in correctly labeled metaphoric instances, is comparable between SLUICE and MTL-SEQ. For example, "covering" in "So who's covering tomorrow?" is wrongly classified as literal by both STL approaches, but correctly labeled as metaphoric by both MTL-all configurations, i.e., with the help of auxiliary tasks. However, most of the newly identified metaphors (i.e., found by MTL-all but not by STL) differ between the approaches. So, in contrast to the first example, "sweet" in "That's a sweet little village" is misclassified by both networks in STL configuration, but correctly labeled as metaphoric only using SLUICE in the MTL-all setting.

Finally, we compare to the state-of-the-art (SOA) on our chosen tasks. We note that such a comparison with SOA is difficult for most tasks due to non-described test splits or usage of cross validation. The latter was too costly in terms of computation time for our tested architectures. Nonetheless, we include reference numbers for an approximate comparison. `adj-noun-met` is the only dataset for which we have the original test split. Here, the original feature based implementation of Tsvetkov et al. (2014) ($F_1 = 0.85$) outperforms our approach (MTL: 0.63) by a large margin. We attribute this to heavy feature-engineering on their part, using supersenses and concreteness information. In contrast, we perform on par with the state-of-the-art on `tok-met` (Do Dinh and Gurevych (2016): $F_1 = 0.56$, our system:

$F_1 = 0.56$). They implement a simple MLP, incorporating also POS tags and concreteness features. Their test set is similarly large as ours. Horbach et al. (2016) report only accuracy (A = 0.86) for their cross validation experiments on `inf-verb`. Our system results are comparable to their approach, albeit in a different setup using a dedicated test set (A = 0.85 for MTL-de). However, as described in Section 2, Horbach et al. (2016) employ a multitude of semantic features, including selectional preferences and topic information. For `part-verb`, we do not reach the results of Köper and Schulte im Walde (2017). They attain $F_1 = 0.88$ using multi-sense embeddings with Multinomial Naive Bayes in a cross validation setting (vs. our $F_1 = 0.81$ on a test set). Since the multi-sense embeddings vastly outperform their own single-sense baseline, it would be interesting for future work to also include this approach into our model.

6 Conclusion

To the best of our knowledge, we are the first to abstract from related non-literal language detection tasks metaphor and idiom classification to a more general model, using multi-task learning. To this end, we presented an evaluation of two different multi-task learning models on four related semantic, non-literal language tasks in English and German: detection of metaphoric tokens, classification of metaphoric adjective-noun constructions, classification of idiomatic use of infinitive-verb compounds, and non-literal particle verbs. We compared both performances to respective single-task learning baselines, and baselines which use merged training data. While results for the latter systems are mixed, the multi-task learning setups improved performance over the baselines in all cases. Especially the metaphor datasets profit substantially from the multi-task setups, and the inclusion of auxiliary tasks—in the case of hard parameter sharing even for out-of-language idiom tasks.

For all but the smallest dataset, soft parameter sharing and learned architecture outperformed hard parameter sharing. However, the latter approach could benefit from more information (using all related tasks as auxiliary tasks), while the former performed best in a mono-lingual setting. In both cases, data from related non-literal language tasks increase classifier performance, which means that a common representation within the network could be established.

Future work could explore more specifically which non-literal language tasks benefit from the inclusion of which auxiliary tasks. Further, it could be investigated how classic features such as the concreteness or supersenses can increase performance when viewed as lower-level semantic auxiliary tasks.

Acknowledgements

This work has been supported by the German Federal Ministry of Education and Research (BMBF) under the promotional reference 01UG1816B (CEDIFOR).

References

Héctor Martínez Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *Proceedings of EACL 2017*, pages 44–53, Valencia, Spain. Association for Computational Linguistics.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Beata Beigman Klebanov, Chee Wee Leong, Michael Heilman, and Michael Flor. 2014. Different texts, same metaphors: Unigrams and beyond. In *Proceedings of the Second Workshop on Metaphor in NLP*, pages 11–17, Baltimore, MD, USA. Association for Computational Linguistics.

The BNC Consortium. 2007. The British National Corpus, version 3 (BNC XML Edition).

Marcel Bollmann and Anders Søgaard. 2016. Improving historical spelling normalization with bi-directional lstms and multi-task learning. In *Proceedings of COLING 2016*, pages 131–139. The COLING 2016 Organizing Committee.

Rich Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48, Amherst, MA, USA. Morgan Kaufmann.

Erik-Lân Do Dinh and Iryna Gurevych. 2016. Token-level metaphor detection using neural networks. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 28–33, San Diego, CA, USA. Association for Computational Linguistics.

Dan Fass. 1991. met *: A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 1(1):49–90.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *Proceedings of EMNLP 2017*, pages 1923–1933, Copenhagen, Denmark. Association for Computational Linguistics.

Andrea Horbach, Andrea Hensler, Sabine Krome, Jakob Prange, Werner Scholze-Stubenrecht, Diana Steffen, Stefan Thater, Christian Wellner, and Manfred Pinkal. 2016. A corpus of literal and idiomatic uses of german infinitive-verb compounds. In *Proceedings of LREC 2016*, pages 836–841, Portorož, Slovenia. European Language Resources Association.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlý, and Vít Suchomel. 2013. The tenten corpus family. In *7th International Corpus Linguistics Conference CL*, pages 125–127, Lancaster, UK.

Tobias Kahse. 2017. Multi-task learning for argumentation mining. Master's thesis, TU Darmstadt.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the Tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.

Maximilian Köper and Sabine Schulte im Walde. 2016. Distinguishing literal and non-literal usage of german particle verbs. In *Proceedings of NAACL-HLT 2016*, pages 353–362, San Diego, CA, USA. Association for Computational Linguistics.

Maximilian Köper and Sabine Schulte im Walde. 2017. Applying multi-sense embeddings for german verbs to determine semantic relatedness and to detect non-literal language. In *Proceedings of EACL 2017*, pages 535–542, Valencia, Spain. Association for Computational Linguistics.

George Lakoff and Mark Johnson. 1980. *Metaphors we live by*. University of Chicago Press, Chicago, IL, USA.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Bilingual word representations with monolingual quality in mind. In *NAACL Workshop on Vector Space Modeling for NLP*, pages 151–159, Denver, CO, US. Association for Computational Linguistics.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33, Berlin, Germany. Association for Computational Linguistics.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. *arXiv preprint, arXiv:1704.06855*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice networks: Learning what to share between loosely related tasks. *arXiv preprint, arXiv:1705.08142*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint, arXiv:1706.05098*.

Marc Schulder and Eduard Hovy. 2014. Metaphor detection through term relevance. In *Proceedings of the Second Workshop on Metaphor in NLP*, pages 18–26, Baltimore, MD, USA. Association for Computational Linguistics.

Claudia Schulz, Steffen Eger, and Johannes Daxenberger. 2018. Multi-task learning for argumentation mining in low-resource settings. In *Proceedings of NAACL-HLT 2018*, pages 35–41, New Orleans, LA, USA. Association for Computational Linguistics.

Roland Schäfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In *Proceedings of LREC 2012*, pages 486–493, Istanbul, Turkey.

Roland Schäfer. 2015. Processing and querying large web corpora with the cow14 architecture. In Piotr Bański, Hanno Biber, Evelyn Breiteneder, Marc Kupietz, Harald Lüngen, and Andreas Witt, editors, *Proceedings of the 3rd Workshop on Challenges in the Management of Large Corpora*, pages 28–34, Lancaster. IDS.

Ekaterina Shutova. 2015. Design and evaluation of metaphor processing systems. *Computational Linguistics*, 41(4):579–623.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of ACL 2016*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Caroline Sporleder and Linlin Li. 2009. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of EACL 2009*, pages 754–762, Athens, Greece. Association for Computational Linguistics.

Gerard J. Steen, Aletta G. Dorst, J. Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A Method for Linguistic Metaphor Identification. From MIP to MIPVU*. John Benjamins, Amsterdam, Netherlands.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of ACL 2014*, pages 248–258, Baltimore, MD, USA. Association for Computational Linguistics.

Yorick Wilks. 1978. Making preferences more active. *Artificial Intelligence*, 11(3):197–223.

Wei Zhang and Judith Gelernter. 2015. Exploring metaphorical senses and word representations for identifying metonyms. *arXiv preprint, arXiv:1508.04515*.