Multi-task Active Learning for Pre-trained Transformer-based Models

Guy Rotman and **Roi Reichart**
Faculty of Industrial Engineering and Management, Technion, IIT, Israel
grotman@campus.technion.ac.il
roiri@technion.ac.il

Abstract

Multi-task learning, in which several tasks are jointly learned by a single model, allows NLP models to share information from multiple annotations and may facilitate better predictions when the tasks are inter-related. This technique, however, requires annotating the same text with multiple annotation schemes, which may be costly and laborious. Active learning (AL) has been demonstrated to optimize annotation processes by iteratively selecting unlabeled examples whose annotation is most valuable for the NLP model. Yet, multi-task active learning (MT-AL) has not been applied to state-of-the-art pre-trained Transformer-based NLP models. This paper aims to close this gap. We explore various multi-task selection criteria in three realistic multi-task scenarios, reflecting different relations between the participating tasks, and demonstrate the effectiveness of multi-task compared to single-task selection. Our results suggest that MT-AL can be effectively used in order to minimize annotation efforts for multi-task NLP models.[1]

1 Introduction

Deep neural networks (DNNs) have recently achieved state-of-the-art results for many natural language processing (NLP) tasks and applications. Of particular importance are contextualized embedding models (McCann et al., 2017; Peters et al., 2018), most of which implement Transformer-based architectures with the self-attention mechanism (Vaswani et al., 2017; Devlin et al., 2019; Raffel et al., 2020).

Nevertheless, DNNs often require large labeled training sets in order to achieve good performance. While annotating such training sets is costly and laborious, the active learning (AL) paradigm aims to minimize these costs by iteratively selecting valuable training examples for annotation. Recently, AL has been shown effective for DNNs across various NLP tasks (Duong et al., 2018; Peris and Casacuberta, 2018; Ein-Dor et al., 2020).

An appealing capability of DNNs is performing multi-task learning (MTL): Learning multiple tasks by a single model (Ruder, 2017). This stems from their architectural flexibility—constructing increasingly deeper and wider architectures from basic building blocks—and in their gradient-based optimization, which allows them to jointly update parameters from multiple task-based objectives. Indeed, MTL has become ubiquitous in NLP (Luan et al., 2018; Liu et al., 2019a).

MTL models for NLP can often benefit from using corpora annotated for multiple tasks, particularly when these tasks are closely related and can inform each other. Prominent examples of multi-task corpora include OntoNotes (Hovy et al., 2006), the Universal Dependencies Bank (Nivre et al., 2020), and STREUSLE (Schneider et al., 2018). Given the importance of multi-task corpora for many MTL setups, effective AL frameworks that support MTL are becoming crucial.

Unfortunately, most AL methods do not support annotations for more than one task. Multi-task AL (MT-AL) was proposed by Reichart et al. (2008) before the neural era, and adapted by Ikhwantri et al. (2018) to a neural architecture. Recently, Zhu et al. (2020) proposed an MT-AL model for slot filling and intent detection, focusing mostly on LSTMs (Hochreiter and Schmidhuber, 1997).

In this paper, we are the first to systematically explore MT-AL for large pre-trained Transformer models. Naturally, our focus is on closely related NLP tasks, for which multi-task annotation of the same corpus is likely to be of benefit. Particularly, we consider three challenging real-life multi-task scenarios, reflecting different relations between the participating NLP tasks: 1. Complementing tasks, where each task may provide

[1]Our code base is available at: https://github.com/rotmanguy/MTAL.

Transactions of the Association for Computational Linguistics, vol. 10, pp. 1209–1228, 2022. https://doi.org/10.1162/tacl_a_00515
Action Editor: Yue Zhang. Submission batch: 5/2021; Revision batch: 7/2022; Published 11/2022.

valuable information to the other task: Dependency parsing (DP) and named entity recognition (NER); 2. Hierarchically related tasks, where one of the tasks depends on the output of the other: Relation extraction (RE) and NER; and 3. Tasks with different annotation granularity: Slot filling (SF, token level) and intent detection (ID, sentence level). We propose various novel MT-AL methods and tailor them to the specific properties of the scenarios in order to properly address the underlying relations between the participating tasks. Our experimental results highlight a large number of patterns that can guide NLP researchers when annotating corpora with multiple annotation schemes using the AL paradigm.

2 Previous Work

This paper addresses a previously unexplored problem: multi-task AL (MT-AL) for NLP with pre-trained Transformer-based models. We hence start by covering AL in NLP and then proceed with multi-task learning (MTL) in NLP.

2.1 Active Learning in NLP

AL has been successfully applied to a variety of NLP tasks, including semantic parsing (Duong et al., 2018), syntactic parsing (Reichart and Rappoport, 2009; Li et al., 2016), co-reference resolution (Li et al., 2020), named entity recognition (Shen et al., 2017), and machine translation (Haffari et al., 2009), to name a few. Recent works demonstrated that models like BERT can benefit from AL in low-resource settings (Ein-Dor et al., 2020; Grießhaber et al., 2020), and Bai et al. (2020) suggested basing the AL selection criterion on linguistic knowledge captured by BERT. Other work performed cost-sensitive AL, where instances may have different costs (Tomanek and Hahn, 2010; Xie et al., 2018). However, most previous work did not apply AL for MTL, which is our main focus.

2.2 Multi-task Learning in NLP

MTL has become increasingly popular in NLP, particularly when the solved tasks are closely related (Chen et al., 2018; Safi Samghabadi et al., 2020; Zhao et al., 2020). In some cases, the MTL model is trained in a hierarchical fashion, where information is propagated from lower-level (sometimes auxiliary) tasks to higher-level tasks (Søgaard and Goldberg, 2016; Rotman and Reichart, 2019; Sanh et al., 2019; Wiatrak and Iso-Sipila, 2020). In other cases, different labeled corpora can be merged to serve as multi-task benchmarks (McCann et al., 2017; Wang et al., 2018). This way, a single MTL model can be trained on multiple tasks, which are typically only distantly related. This research considers the setup of closely related tasks where annotating a single corpus w.r.t. multiple tasks is a useful strategy.

3 Task Definition - Multi-task Active Learning

In the MT-AL setup, the AL algorithm is provided with a textual corpus, where an initial (typically small) set of n_0 examples is labeled for t tasks. The AL algorithm implements an iterative process, where at the i-th iteration the goal of the AL algorithm is to select n_i additional unlabeled examples that will be annotated on all t tasks, such that the performance of the base NLP model will be improved as much as possible with respect to all of them. While such a greedy strategy of gaining the most in the i-th iteration may not yield the best performance in subsequent iterations, most AL algorithms are greedy, and we hence follow this strategy here as well.

We focus on the standard setup of confidence-based AL, where unlabeled examples with the lowest model confidence are selected for annotation. Algorithm 1 presents a general sketch of such AL algorithms, in the context of MTL. This framework, first introduced by Reichart et al. (2008), is a simple generalization of the single-task AL (ST-AL) framework, which supports the annotation of data with respect to multiple tasks.

Algorithm 1 Multi-task Confidence-based Active Learning (Confidence-based MT-AL)

```
Input: Labeled data L (annotated on $t$ tasks), Unlabeled data U
Algorithm:
For $i = 1, \dots, \mathcal{T}$:
    1. Train a multi-task learning (MTL) model $h$ on L.
    2. For each $u \in$ U calculate its aggregated confidence
       score $C_h(u)$ on all $t$ tasks according to $h$.
    3. Choose the $n_i$ unlabeled examples from U with the
       lowest confidence score $C_h(u)$ and send them for
       annotation according to all $t$ tasks.
    4. Add the newly labeled examples to L and remove them
       from U.
```

Selection Method	Description	Participating Tasks for Training	Participating Tasks for Selection
ST-R	Single-task random selection	One	None
ST-EC	Single-task entropy-based confidence	One	One
ST-DA	Single-task dropout agreement	One	One
MT-R	Multi-task random selection	All	None
MT-EC	Multi-task entropy-based confidence	All	One
MT-DA	Multi-task dropout agreement	All	One
MT-AVG	Multi-task average entropy-based confidence	All	All
MT-AVGDA	Multi-task average dropout agreement	All	All
MT-MAX	Multi-task maximum entropy-based confidence	All	All
MT-MIN	Multi-task minimum entropy-based confidence	All	All
MT-PAR	Multi-task Pareto entropy-based confidence	All	All
MT-RRF	Multi-task Reciprocal Rank Fusion entropy-based confidence	All	All
MT-IND	Multi-task independent selection entropy-based confidence	All	All

Table 1: Summary of the ST-AL and MT-AL selection methods explored in this paper.

As discussed in §1, we explore several variations of the MT-AL setup: Independent tasks that inform each other (§5), hierarchically related tasks, where one task depends on the output of the other (§6), and tasks with different annotation granularity: word- and sentence-level (§7). Before we can introduce the MT-AL algorithms for each of these setups, we first need to lay their shared foundations: The single-task and multi-task model confidence scores.

4 Confidence Estimation in Single-task and Multi-task Active Learning

We now introduce the confidence scores that we consider for single-task (ST-AL) and multi-task (MT-AL) active learning. These confidence scores are essentially the core of confidence-based AL algorithms (see Steps 2-3 of Algorithm 1). In Table 1 we provide a summary of the various ST-AL and MT-AL selection methods we explore.

4.1 Single-task Confidence Scores

We consider three confidence scores that have been widely used in ST-AL:

Random (*ST-R*) This baseline method simply assigns random scores to the unlabeled examples.

Entropy-based Confidence (*ST-EC*) The single-task entropy-based confidence score is defined as:

$$\textit{ST-EC}(x) = 1 - E(x). \qquad (1)$$

For sentence classification tasks such as ID, E(x) is simply the entropy over the class predictions of a sample x divided by the log number of labels. In our token classification tasks (DP, NER, RE, and SF), E(x) is the normalized sentence-level entropy (Kim et al., 2006), which allows us to estimate the uncertainty of the model for a given sequence of tokens $x = (x_1 \dots x_m)$:

$$E(x) = -\frac{1}{m \cdot \log s} \sum_{i=1}^{m} \sum_{j=1}^{s} p(y_j|x_i) \log p(y_j|x_i), \qquad (2)$$

where m is the number of tokens, y_j is the j'th possible label, and s is the number of labels. We perform entropy normalization by averaging the token-level entropies, in order to mitigate the effect of the sentence length, and by dividing the score by the log number of labels. The resulting confidence score ranges from 0 to 1, where lower values indicate lower certainty.

Dropout Agreement (*ST-DA*) Ensemble methods have proven effective for AL (see, e.g., Seung et al., 1992; Settles and Craven, 2008). In this paper, we derive a confidence score inspired by Reichart and Rappoport (2007). We start by creating $k = 10$ different models by performing *dropout inference* for k times (Gal and Ghahramani, 2016). We then compute the single-task dropout agreement score for a sentence x by calculating the average token-level agreement across model pairs:

$$\textit{ST-DA}(x) = \frac{1}{m \cdot k \cdot (k-1)} \sum_{j \neq j'} \sum_{i=1}^{m} {}_{\{\hat{y}_i^j = \hat{y}_i^{j'}\}}, \qquad (3)$$

where $\hat{y}_i^j$ is the predicted label of model j for the i'th token. The resulting scores range from 0 to 1, where lower values indicate lower certainty.[2]

[2]For sentence classification tasks, *ST-DA* is computed similarly, without averaging over the tokens.

4.2 Multi-task Confidence Scores

When deriving confidence scores for MT-AL, multiple design choices should be made. First, the confidence score of a multi-task model can be based on both tasks or only on one of them. We denote with ***MT-EC*** and ***MT-DA*** the confidence scores that are equivalent to *ST-EC* and *ST-DA*: The only (important) difference is that they are calculated for a multi-task model. For clarity, we will augment this notation with the name of the task according to which the confidence is calculated. For example, *ST-EC-NER* and *MT-EC-NER* are the EC scores calculated using the named entity recognition (NER) classifier of a single-task and a multi-task model, respectively.

We can hence evaluate MT-AL algorithms on cross-task selection, that is, when the evaluated task is different from the task used for computing the confidence scores (and hence for sample selection). For example, evaluating the performance of a multi-task model, trained jointly on NER and DP, on the DP task when the confidence scores used by the MT-AL algorithm are only based on the NER classifier (*MT-EC-NER*).

We also consider a family of confidence scores for MT-AL that are computed with respect to all participating tasks (joint-selection scores). For this aim, we consider three simple aggregation schemes using the average, maximum, or minimum operators over the single-task confidence scores. For example, the multi-task average confidence (***MT-AVG***) averages for a sample x the entropy-based confidence scores over all t tasks:

$$MT\text{-}AVG(x) = \frac{1}{t}\sum_{i=1}^{t} MT\text{-}EC\text{-}i(x), \qquad (4)$$

The multi-task average dropout agreement score (***MT-AVGDA***) is similarly defined, but the averaging is over the *MT-DA* scores. Finally, the multi-task maximum (minimum) ***MT-MAX*** (***MT-MIN***) is computed in a similar manner to *MT-AVG* but with the $\max$ ($\min$) operator taken over the task-specific confidence entropies.

Beyond Direct Manipulations of Confidence Scores Since our focus in this paper is on multi-task selection, we would like to consider additional selection methods which go beyond the simple methods in previous work. The common principle of these methods is that they are less sensitive to the actual values of the confidence scores and instead consider the relative importance of the example to the participating tasks.

First, we consider ***MT-PAR***, which is based on the Pareto-efficient frontier (Lotov and Miettinen, 2008). We start by representing each unlabeled sample as a t-dimensional space vector c, where $c_i =$ *MT-EC-i* is the MT confidence score for task i. Next, we select all samples for which the corresponding vector belongs to the Pareto-efficient frontier. A point belongs to the frontier if for every other vector c' the following holds: 1. $\forall i \in [t], c_i \leq c'_i$ and 2. $\exists i \in [t], c_i < c'_i$. If the number of samples in the frontier is smaller than the total number of samples to select (n), we re-iterate the procedure by removing the vectors of the selected samples and calculating the next Pareto points. If there are still p points to be selected but the number of the final Pareto points (f) exceeds p, we select every $\lfloor \frac{f}{p} \rfloor$ point, ordered by the first axis.

Next, inspired by the field of *information retrieval*, we propose ***MT-RRF***. This method allows us to consider the rank of each example with respect to the participating tasks, rather than the actual confidence values. We first calculate r_i, the ranked list of the i-th task, by ranking the examples according to their *MT-EC-i* scores, from lowest to highest. We next fuse the resulting t ranked lists into a single ranked list R, using the reciprocal rank fusion (RRF) technique (Cormack et al., 2009). The RRF score of an example x is computed as:

$$RRF\text{-}Score(x) = \sum_{i=1}^{t} \frac{1}{k + r_i(x)}, \qquad (5)$$

where k is a constant, set to 60, as in the original paper. The final ranking is computed over the RRF scores of the examples—from highest to lowest. Higher-ranked examples are chosen first for annotation as they have lower confidence scores. Finally, ***MT-IND*** independently selects the $\lfloor \frac{n}{t} \rfloor$ most uncertain samples according to each task by ranking the *MT-EC* scores and re-iterating if overlaps occur.

We finally compare the selected samples of six of the MT-AL methods, after training a multi-task model for a single AL iteration on the DP and NER tasks (Figure 1). It turns out that while some of the methods tend to choose very similar example subsets for annotation (e.g., *MT-IND* and *MT-RRF* share 94% of the selected samples,

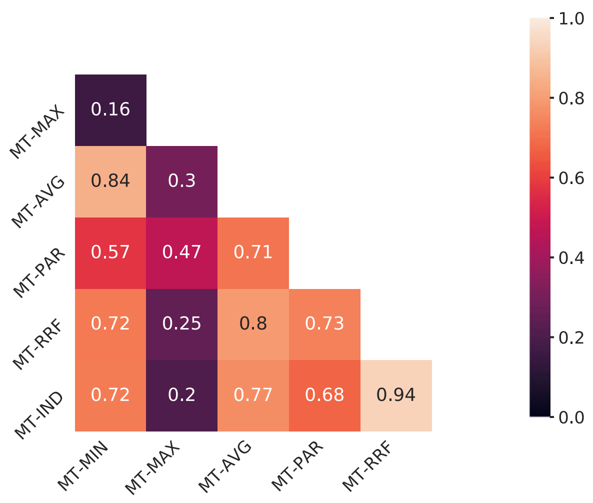


Figure 1: The percentage of shared selected samples between pairs of MT-AL selection methods (see experimental details in the text).

and *MT-AVG* and *MT-MIN* share 84% of them), others substantially differ in their selection (e.g., *MT-MAX* shares only 16% of its selected samples with *MT-MIN* and 20% with *MT-IND*). This observation encourages us to continue investigating the impact of the various selection methods on MT-AL.

5 MT-AL for Complementing Tasks

We start by investigating MT-AL for two closely related, complementing, syntactic tasks: Dependency Parsing (DP) and Named Entity Recognition (NER), which are often solved together by a joint multi-task model (Finkel and Manning, 2009; Nguyen and Nguyen, 2021).

5.1 Research Questions

We focus on three research questions. At first, we would like to establish whether MT-AL methods are superior to ST-AL methods for multi-task learning. Our first two questions are hence: **Q1.1**: Is multi-task learning effective in this setup? and **Q1.2**: Is AL effective? If so, which AL strategy is better: ST-AL or MT-AL?

Next, notice that in MT-AL the confidence score of an example can be based on one or more of the participating tasks. That is, even if the base model for which training examples are selected is an MTL model, the confidence scores used by the MT-AL algorithm can be based on one task or more (§4.2). Our third question is thus: **Q1.3**: Is it better to calculate confidence scores based on one of the participating tasks, or should we consider a joint confidence score, based on both tasks?[3]

5.2 Data

We consider the English version of the OntoNotes 5.0 corpus (Hovy et al., 2006), consisting of seven textual domains: broadcast conversation (BC), broadcast news (BN), magazine (MZ), news (NW), bible (PT), telephone conversation (TC), and web (WB). Sentences are annotated with constituency-parse trees, named entities, part-of-speech tags, as well as other labels. We convert constituency-parse trees to dependency trees using the ElitCloud conversion tool.[4] We do not report results in the PT domain, as it is not annotated for NER. Table 2 summarizes the number of sentences per split for the OntoNotes domains, as well as for the additional datasets used in our next setups.

5.3 Models

We consider two model types: Single-task and multi-task models. Our single-task model (ST) consists of the 12-layer pre-trained BERT-base encoder (Devlin et al., 2019), followed by a task decoder. At first, we implemented a simple multi-task model (SMT), consisting of a shared 12-layer pre-trained BERT-base encoder followed by an independent decoder for each task. However, early results suggested that it is inferior to single-task modeling. We therefore implemented a more complex multi-task model (CMT), illustrated in Figure 2. This model consists of (shared) cross-task and task-specific modules, similar in nature to the architecture proposed by Lin et al. (2018). In particular, it uses the 8 bottom BERT layers as shared cross-task layers and employs $t+1$ replications of the 4 top BERT layers, one replication for each task, as well as a shared cross-task replication. The input text, as encoded by the shared 8 layers, $e^{S}_{1:8}$, is passed through the shared and non-shared 4-layer modules, $e^{S}_{8:12}$ and $e^{U^i}_{8:12}$, respectively. The task classifiers are then fed with the output of the cross-task layers combined with the output of their task-specific layers,

[3]This question naturally generalizes when more than two tasks are involved.

[4]`https://github.com/elitcloud/elit-java`.

	DP-NER						NER-RE					SF-ID	
	BC	BN	MZ	NW	TC	WB	NYT24	NYT29	ScieRC	WebNLG	WLP	ATIS	SNIPS
Train	11,877	10,681	6,771	34,967	12,889	15,639	56,193	63,305	1,540	4,973	6,690	4,478	13,084
Dev	2,115	1,293	640	5,894	1,632	2,264	5,000	7,033	217	500	2,320	500	700
Test	2,209	1,355	778	2,325	1,364	1,683	5,000	4,006	451	689	2,343	893	700

Table 2: Data statistics. We report the number of sentences in the original splits for each pair of tasks.

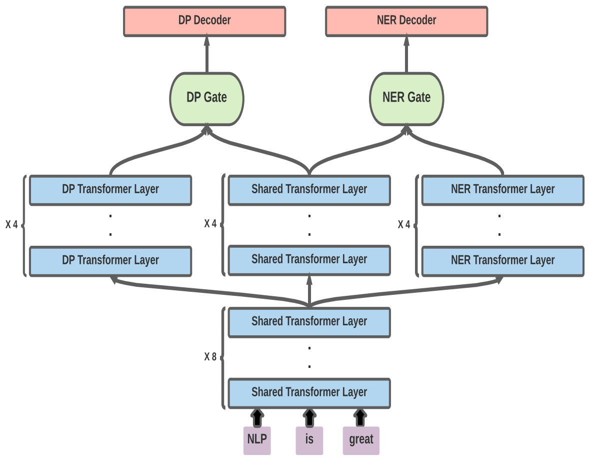


Figure 2: Our complex multi-task model architecture for DP and NER.

following the gating mechanism of Rotman and Reichart (2019):

$$a^i(x) = \sigma(W_g^i[e_{8:12}^S(x); e_{8:12}^{U^i}(x)] + b_g^i),$$
$$g^i(x) = a^i(x) \odot e_{8:12}^S(x) + (1 - a^i(x)) \odot e_{8:12}^{U^i}(x),$$

where ; is the concatenation operator, $\odot$ is the element-wise product, σ is the Sigmoid function, and W_g^i and b_g^i are the gating mechanism parameters. The combined vector $g^i(x)$ is then fed to the i-th task-specific decoder.[5]

All implementations are based on HuggingFace's Transformers package (Wolf et al., 2020).[6] For all models, the DP decoder is based on the Biaffine parser (Dozat and Manning, 2017) and the NER decoder is a simple linear classifier.

5.4 Training and Hyper-parameter Tuning

We consider the following hyper-parameters for the AL experiments. At first, we randomly sample 2% of the original training set to serve as the initial labeled examples in all experiments and treat the rest of the training examples as unlabeled. We also fix our development set to be twice the size of our initial training set, by randomly sampling examples from the original development set. We then run each AL method for 5 iterations, where at each iteration, the algorithm selects an unlabeled set of the size of its initial training set (that is, 2% of the original training set) for annotation. We then reveal the labels of the selected examples and add them to the training set of the next iteration. At the beginning of the final iteration, our labeled training set consists of 10% of the original training data.

In each iteration, we train the models with 20K gradient steps with an early stopping criterion according to the development set. We report LAS scores for DP and F1 scores for NER. For DP, we measure our AL confidence scores on the unlabeled edges. When performing multi-task learning, we set the stopping criterion as the geometric mean of the task scores (F1 for NER and LAS for DP). We optimize all parameters using the ADAM optimizer (Kingma and Ba, 2015) with a weight decay of 0.01, a learning rate of 5e-5, and a batch size of 32. For label smoothing (see below), we use $\alpha = 0.2$. Following Dror et al. (2018), we use the t-test for measuring statistical significance (*p-value* $= 0.05$).

5.5 Results

Model Architecture (Q1.1) We would first like to investigate the performance of the single-task and multi-task models in the full training (FT) and, more importantly, in the active learning (AL) setups. We hence compare three architectures: The single-task model (ST), the simple multi-task model (SMT), and our complex multi-task model (CMT). We train each model for DP and for NER on the six OntoNotes domains using the cross-entropy (CE) objective function, or with the label smoothing objective (LS (Szegedy et al., 2016)) that has been demonstrated to decrease calibration errors of Transformer models (Desai

[5]We considered several other parameter-sharing schemes but witnessed lower performance.

[6]`https://github.com/huggingface/transformers`.

	Full Training				Active Learning			
	DP		NER		DP		NER	
	Avg	Best	Avg	Best	Avg	Best	Avg	Best
ST (CE)	**87.17**	1/6	70.35	0/6	**86.43**	**3/6**	**74.65**	6/6
SMT (CE)	86.94	2/6	67.51	0/6	85.86	2/6	70.31	0/6
CMT (CE)	87.04	**3/6**	**72.79**	**6/6**	85.91	1/6	72.11	0/6
ST (LS)	87.64	0/6	71.31	1/6	88.96	0/6	**75.61**	2/6
SMT (LS)	86.87	0/6	69.07	0/6	87.53	0/6	73.26	1/6
CMT (LS)	**87.98**	**6/6**	**72.86**	**5/6**	**89.03**	**6/6**	74.44	**3/6**

Table 3: A comparison of a single-task model (ST), a simple multi-task model (SMT), and our complex multi-task model (CMT) in full training and active learning. Models were trained with the cross-entropy (CE) or label smoothing (LS) losses, on all OntoNotes domains.

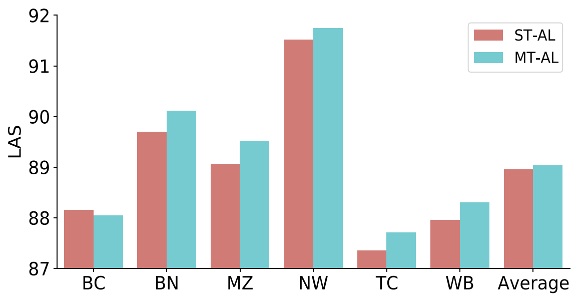
and Durrett, 2020; Kong et al., 2020). ST-AL is performed with *ST-EC* and MT-AL with *MT-AVG*.

Table 3 reports the average scores (Avg column) over all domains and the number of domains where each model achieved the best results (Best). The results raise three important observations. First, SMT is worse on average than ST in all setups, suggesting that ***vanilla MT is not always better than ST training***. Second, ***our CMT model achieves the best scores in most cases***. The only case where it is inferior to ST, but not to SMT, is on AL with CE training. However, when training with LS, it achieves results comparable to or higher than those of ST on AL.

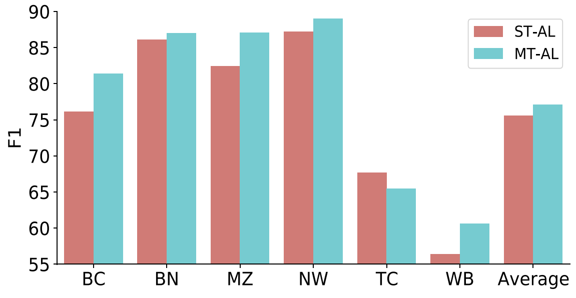
Third, when comparing CE to LS training, ***LS clearly improves the average scores of all models*** (besides one case). Interestingly, the improvement is more significant in the AL setup than in the FT setup. We report that when expanding these experiments to all AL selection methods, LS was found very effective for both tasks, outperforming CE in most comparisons, with an average improvement of 1.8% LAS for DP and 0.9 F1 for NER.

Multi-task vs. Single-task Performance (Q1.2) We next ask whether MT-AL outperforms strong ST-AL baselines. Figure 3 presents for every task and domain the performance of the per-domain best ST-AL and MT-AL methods after the final AL iteration. Following our observations in Q1.1, we train all models with the LS objective and base the multi-task models on the effective CMT model.

Although there is no single method, MT-AL or ST-AL, which performs best across all domains and tasks, MT-AL seems to perform consistently



(a) Dependency Parsing



(b) Named Entity Recognition

Figure 3: Performance of the best ST-AL vs. the best MT-AL method per domain (Q1.2).

better. The figure suggests that ***MT-AL is effective for both tasks, outperforming the best ST-AL methods in 4 of 6 DP domains*** (results are not statistically significant, the average *p-value* is 0.19) ***and in 5 of 6 NER domains*** (results for 3 domains are statistically significant). While the average gap between MT-AL and ST-AL is small for DP (0.28% LAS), in NER it is as high as 2.4 F1 points in favor of MT-AL. In fact, for half of the NER domains, this gap is greater than 4.2 F1.

When comparing individual selection methods, *MT-AVG*, and multi-task DP-based entropy, *MT-EC-DP*, are the best selection methods for DP, with average scores of 89.03% and 88.99%, respectively. Single-task DP-based entropy, *ST-EC-DP*, is third, with an average score of 88.96%, while the second best ST-AL method, *ST-DA-DP*, is ranked only ninth among all methods, outperformed by seven different MT-AL methods. For NER, multi-task NER-based entropy, *MT-EC-NER*, is the best model with an average F1 score of 77.13, followed by *MT-MAX* with an average F1 score of 75.90. The single-task NER-based methods, *ST-EC-NER* and *ST-DA-NER* are ranked only fifth and sixth both with an average score of 75.60. These results provide an additional indication of the superiority of MT-AL.

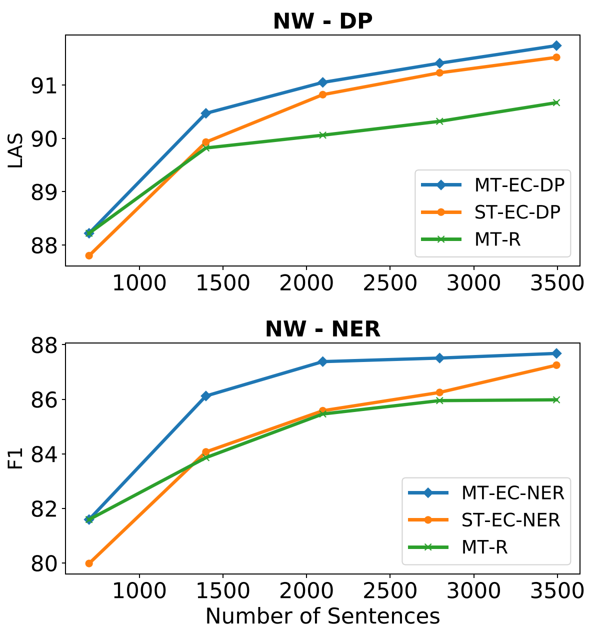


Figure 4: Performance as a function of the number of training examples (Q1.2).

In terms of the MT-AL selection methods that do not perform a simple aggregation, *MT-PAR* and *MT-RRF* perform similarly, averaging 88.31% LAS for DP and 75.88 F1 for NER, while *MT-IND* achieves poor results for DP and moderate results for NER (an overall comparison of the MT-AL methods is provided in § 8).

We next compare the per-iteration performance of ST-AL and MT-AL. To this end, Figure 4 presents the performance for the most prominent MT-AL and ST-AL methods: *MT-EC-DP* and *ST-EC-DP* for DP and *MT-EC-NER* and *ST-EC-NER* for NER, together with the multi-task random selection method *MT-R*. We plot for each method its task score on the NW domain (the one with the largest dataset) as a function of the training set size, corresponding to 2% to 10% of the original training examples. ***Clearly, the MT-AL methods are superior across all AL iterations, indicating the stability of MT-AL as well as its effectiveness in low-resource setups***. Similar patterns are also observed in the other domains.

As a final evaluation for Q1.2, we directly compare pairs of MT-AL and ST-AL methods, performing three comparison types on each of the domains: **Within-task**: Comparing the performance of *ST-EC-i* and *ST-DA-i* to their MT-AL counterparts (*MT-EC-i* and *MT-DA-i*) and to the joint-selection methods on task *i*, either DP or NER (108 comparisons); **Cross-task**: Comparing the performance of *ST-EC-i* and *ST-DA-i* to their MT-AL counterparts and to the joint-selection methods on the opposite task (e.g., if the models select according to DP, we evaluate the NER task; 108 comparisons). This comparison allows us to evaluate the effect of single- and multi-task modeling on cross-task performance. Since single-task models cannot be directly applied to the opposite task, we record the examples selected by the ST-AL method and train a model for the opposite task on these examples; and **Average**: Comparing all ST-AL methods to all MT-AL methods according to their average performance on both tasks (264 comparisons).

	Within-task		Cross-task		Average
	DP	NER	DP	NER	DP + NER
MT-AL winning %	47.22	63.88	90.74	89.81	78.78

Table 4: A comparison of MT-AL vs. ST-AL on within-task, cross-task, and average performance. Values indicate the percentage of comparisons in which MT-AL methods were superior.

Table 4 reports the percentage of comparisons where the MT-AL methods are superior. On average, the two method types are on par when comparing **Within-task** performance. More interestingly, for **Cross-task** performance MT-AL methods are clearly superior with around 90% winnings (87% of the cases statistically significant). Finally, the **Average** also supports the superiority of MT-AL methods which perform better in 79% of the cases (all results are statistically significant). ***These results demonstrate the superiority of MT-AL, particularly (and perhaps unsurprisingly) when both tasks are considered.***

Single-task vs. Joint-task Selection (Q1.3) Next, we turn to our third question, which compares single-task vs. joint-task confidence scores. That is, we ask whether MT-AL methods that base their selection criterion on more than one task are better than ST-AL and MT-AL methods that compute confidence scores using a single task only.

To answer this question, we compare the two best ST-AL and MT-AL methods that are based on single-task selection to the two best joint-task selection MT-AL methods. As previously, all methods employ the LS objective. Table 5 reports the average scores (across domains) of each

	DP	NER	Average
ST-EC-DP	88.96	71.34	80.15
ST-EC-NER	86.66	75.75	81.21
MT-EC-DP	88.99	73.75	81.37
MT-EC-NER	86.90	**77.13**	82.01
MT-AVG	**89.02**	74.44	81.74
MT-MAX	88.64	75.90	**82.27**

Table 5: A comparison of AL methods that base their selection on a single-task vs. joint-task confidence scores. Results are averaged across the OntoNotes domains (Q1.3).

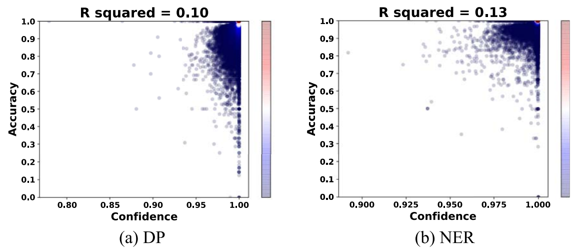


Figure 5: Sentence-level accuracy as a function of entropy-based confidence, for DP (left) and for NER (right), when training with the CE objective. The heat maps represent the point frequency.

of these methods for DP, NER, and the average task score, based on the final AL iteration.

While the method that performs best on average on both tasks is *MT-MAX*, a joint-selection method, the second best method is *MT-EC-NER*, a single-task selection method, and the gap is only 0.26 points. Not surprisingly, performance is higher when the evaluated task also serves as the task that the confidence score is based on, either solely or jointly with another task.

Although the joint-selection methods are effective for both tasks, we cannot decisively conclude that they are better than MT-AL methods that perform single-task selection. However, we do witness another confirmation for our answer to Q1.2, as all presented MT-AL methods perform better on average on both tasks (the *Average* column) than the ST-AL methods.

Overconfidence Analysis Originally, we trained our models with the standard CE loss. However, our early experiments suggested that such CE-based training yields overconfident models, which is likely to severely harm confidence-based AL methods. While previous work demonstrated the positive impact of label smoothing (LS) on model calibration, to the best of our knowledge, the resulting impact on multi-task learning has not been explored, specifically not in the context of AL. We next analyze this impact, which is noticeable in our above results, in more detail.

Figure 5 presents sentence-level confidence scores as a function of sentence-level accuracy when separately training a single-task BERT-base model on DP (left figure) and on NER (right figure) with the CE objective. The confidence scores were computed according to the *ST-EC* scores. The figure confirms that the model tends to be overconfident in its predictions. Furthermore, the low R^2 values (0.1 and 0.13 for DP and NER, respectively) indicate poor model calibration, since confidence scores are not correlated with actual accuracy. Similar patterns were observed when training our multi-task models with the CE objective.

Following this analysis, we turn to investigate the impact of LS on model predictions in MT-AL. Inspired by Thulasidasan et al. (2019), who defined the *overconfidence error* (OE) for classification tasks, we first slightly generalize OE to support sentence-level scores for token classification tasks. Given N sentences, for each sentence x we start by calculating its accuracy score $acc(x)$ over its tokens. The confidence score $conf(x)$ is set to the confidence score of the corresponding AL method. We then define OE as:

$$OE = \frac{1}{N}\sum_{x=1}^{N} conf(x) \times max\big(conf(x) - acc(x), 0\big).$$

In essence, OE penalizes predictions according to the gap between their confidence score and their accuracy, but only when the former is higher.

In Table 6 we compare the OE scores of *ST-EC*, trained with the LS objective to 3 alternatives: *ST-EC* trained with the CE objective, *ST-EC* with the post-processing method *temperature scaling* (TS), and *ST-DA* trained with the CE objective. Both TS and dropout inference have been shown to improve confidence estimates (Guo et al., 2017; Ovadia et al., 2019), and hence serve as alternatives to LS in this comparison. OE scores are reported on the unlabeled set (given the true labels in hindsight) at the final AL iteration for both tasks. Additionally, OE scores for *MT-AVG* and *MT-AVGDA* are also reported and averaged on both tasks.

	DP						
	BC	BN	MZ	NW	TC	WB	Average
ST-EC-DP (CE)	0.0720	0.0672	0.0757	0.0506	0.0584	0.3064	0.1051
ST-EC-DP (TS)	0.0753	0.0651	0.0673	0.0421	0.0545	0.3012	0.1009
ST-EC-DP (LS)	**0.0269**	**0.0186**	**0.0122**	**0.0107**	**0.0289**	0.1725	**0.0450**
ST-DA-DP (CE)	0.0410	0.0392	0.0414	0.0337	0.0381	**0.1098**	0.0505
	NER						
	BC	BN	MZ	NW	TC	WB	Average
ST-EC-NER (CE)	0.0111	0.0167	0.0131	0.0146	0.0090	0.0132	0.0130
ST-EC-NER (TS)	0.0100	0.0172	0.0136	0.0108	0.0087	0.0142	0.0124
ST-EC-NER (LS)	**0.0002**	**0.0002**	**0.0001**	**0.0002**	**0.0009**	**0.0010**	**0.0004**
ST-DA-NER (CE)	0.0090	0.0121	0.0117	0.0122	0.0082	0.0110	0.0107
	DP + NER						
	BC	BN	MZ	NW	TC	WB	Average
MT-AVG (CE)	0.0436	0.0499	0.0473	0.0345	0.0393	0.1681	0.0637
MT-AVG (TS)	0.0447	0.0468	0.0460	0.0318	0.0396	0.1645	0.0622
MT-AVG (LS)	**0.0040**	**0.0018**	**0.0013**	**0.0012**	**0.0057**	**0.0546**	**0.0114**
MT-AVGDA (CE)	0.0291	0.0287	0.0302	0.0236	0.0257	0.0718	0.0349

Table 6: Overconfidence Error Results.

The results are conclusive, ***LS is the least overconfident method, achieving the lowest OE scores on all 18 setups, but one***. While LS achieves a proportionate reduction error (PRE) of between 57.2% and 96.7% compared to the standard CE method, DA achieves at most a PRE of 51.9% and TS seems to have almost no effect. These results confirm that LS is highly effective in reducing overconfidence scores for BERT-based models, and we are able to show for the first time that such a reduction also holds for multi-task models.

6 MT-AL for Hierarchically Related Tasks

Until now, we have considered tasks (DP and NER) that are mutually informative but can be trained independently of each other. However, other multi-task learning scenarios involve a task that is dependent on the output of another task. A prominent example is the relation extraction (RE) task that depends on the output of the NER task, since the goal of RE is to classify and identify relations between named entities. Importantly, if the NER part of the model does not perform well, this harms the RE performance as well. Sample selection in such a setup should hence reflect the hierarchical relation between the tasks.

6.1 Selection Methods

Since the quality of the classifier for the independent task (NER) now affects also the quality of the classifier for the dependent task (RE), the confidence of each of the tasks may get different relative importance values. Although this in principle can also be true for independent tasks (§5), explicitly accounting for this property seems more crucial in the current setup.

We hence modify four of our joint-selection methods (§4) to reflect the inherent a-symmetry between the tasks, by presenting a scaling parameter $0 \leq \beta \leq 1$:[7]

a) *MT-AVG* is now calculated as follows:

$$\begin{aligned} \textit{MT-AVG}(x) = \beta \cdot \textit{MT-EC-RE}(x) + \\ (1-\beta) \cdot \textit{MT-EC-NER}(x). \end{aligned}$$

b) *MT-RRF* is calculated similarly by multiplying the RRF term of RE by β and that of NER by $1-\beta$.

c) *MT-IND* is calculated by independently choosing $100 \cdot \beta\%$ of the selected samples according to the RE scores and $100 \cdot (1-\beta)\%$ according to the NER scores.

d) *MT-PAR* is computed by restricting the first Pareto condition for the position of the RE confidence score: $c_{RE} \leq q_\beta \cdot c'_{RE}$, where q_β is the value of the β-quantile of the RE confidence scores.[8] We apply such a condition if $\beta < 0.5$. Otherwise, if $\beta > 0.5$ the condition is applied to the NER component, and when it is equal to 0.5, the original Pareto method is used. Since we restrict the condition to only one of the tasks, fewer samples will meet this condition (since $0 \leq q_\beta \leq 1$), and the Pareto frontier will include more samples that have met the condition for the second task.

6.2 Research Questions

In our experiments, we would like to explore two research questions: **Q2.1**: Which MT-AL selection methods are most suitable for this setup? and **Q2.2**: What is the best balance between the participating tasks?

Since RE fully relies on the output of NER, we limit our experiments only to joint multi-task models and do not include single-task models.

[7] We do not include the *MT-MAX* and *MT-MIN* methods in our evaluation. Since the two tasks exhibit confidence scores in a similar range, scaling their confidence scores according to these methods resulted in selecting samples based almost solely on the task with the higher (in the case of *MT-MAX*) or lower (in the case of *MT-MIN*) scaling parameter.

[8] The second Pareto condition is similarly modified, but now with the $<$ sign.

	NER	NER Best	RE	RE Best
MT-R	81.96	0/5	52.47	05
MT-AVG ($\beta = 1.0$)	82.64	1/5	59.51	2/5
MT-RRF ($\beta = 0.8$)	82.69	1/5	59.65	1/5
MT-IND ($\beta = 1.0$)	**82.86**	2/5	**60.15**	0/5
MT-PAR ($\beta = 0.7$)	82.68	1/5	59.57	2/5

Table 7: Hierarchical MT-AL results. We report best average F1 results over all five datasets for the best β configuration per method.

6.3 Experimental Setup

We experiment with the span-based joint NER and RE BERT model of Li et al. (2021).[9] Experiments were conducted on five diverse datasets: NYT24 and NYT29 (Nayak and Ng, 2020), ScieRC (Luan et al., 2018), WebNLG (Gardent et al., 2017), and WLP (Kulkarni et al., 2018). The AL setup is identical to that of §5.4. Other hyper-parameters that were not mentioned before are identical to those of the original implementation.

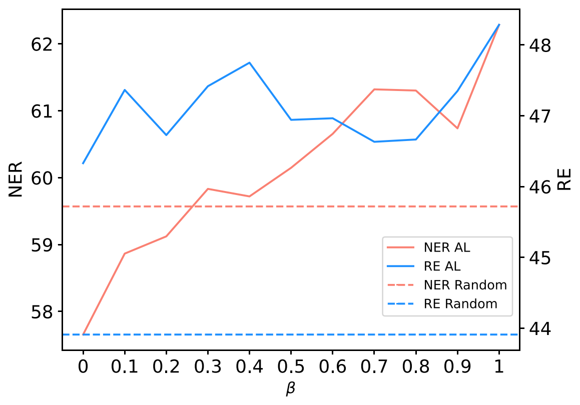
6.4 Results

Best Selection Method (Q2.1) We start by identifying the best selection method for this setup. Table 7 summarizes the per-task average score for the best β value of each method across the five datasets.

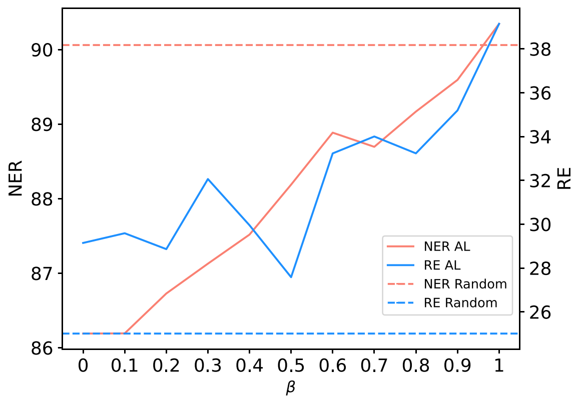
We observe three interesting patterns. First, ***MT-AL is very effective in this setup for the dependent task (RE)***, while for the independent task (NER), random selection does not fall too far behind. Second, ***all MT-AL methods achieve better performance for higher β values*** by giving more weight to the RE confidence scores during the selection process. This is an indication that indeed the selection method should reflect the asymmetric nature of the tasks. Third, overall, ***MT-IND is the best performing method***, averaging first in NER and in RE, while *MT-AVG*, *MT-RRF* and *MT-PAR* achieve similar results in both tasks.

Scaling Configuration (Q2.2) Figure 6 presents the average F1 scores of the four joint-selection methods, as well as the random selection method *MT-R*, as a function of β (the relative weight of the RE confidence). First, we notice that joint selection outperforms random selection in the WebNLG domain only for RE (the dependent task) but not for NER (except when β approaches 1). Second, and more importantly, $\beta = 1$, that is, ***selecting examples only according to the confidence score of the RE (dependent) task, is most beneficial for both tasks*** (Q2.1). We hypothesize that this stems from the fact that the RE confidence score conveys information about both tasks and that this combined information provides a stronger signal with respect to the NER (independent) task, compared to the NER confidence score. Interestingly, the positive impact of higher values of β is more prominent for NER, even though this means that the role of the NER confidence score is downplayed in the sample selection process.

For the individual selection methods, we report that *MT-AVG* and *MT-IND* achieve higher results as β increases, while *MT-PAR* and *MT-RRF* peak at $\beta = 0.7$ and $\beta = 0.8$, respectively, and then drop by 0.2 F1 points for NER and 0.9 F1 points for RE on average.



(a) ScieRC



(b) WebNLG

Figure 6: Average F1 scores over four joint-selection methods as a function of β (the relative weight of the RE confidence).

[9]https://github.com/JiachengLi1995/JointIE.

7 MT-AL for Tasks with Different Annotation Granularity

NLP tasks are defined on different textual units, with the most common examples of sentence-level and token-level tasks. Our last investigation considers the scenario of two closely related tasks that are of different granularity: Slot filling (SF, token-level) and intent detection (ID, sentence-level).

Due to the different annotation nature of the two tasks, we have to define the cost of example annotation with respect to each. Naturally, there is no correct way to quantify these costs, but we aim to propose a realistic model. We denote the cost of annotating a sample for SF with $Cost_{SF} = m + tp \cdot nt$, where m is the number of tokens in the sentence, tp is a fixed token annotation cost (we set $tp = 1$) and nt is the number of entities. The cost of annotating a sample for ID is next denoted with $Cost_{ID} = m + ts$, where ts is a fixed sentence cost (we set $ts = 3$). Our solution (see below) allows some examples to be annotated only with respect to one of the tasks. For examples that are annotated with respect to both tasks we consider an additive joint cost where the token-level term m is considered only once: $JCost = Cost_{SF} + Cost_{ID} - m$. In our experiments, we allow a fixed annotation budget of $B = 500$ per AL iteration.

7.1 Methods

We consider three types of decision methods: **Greedy Active Learning (GRD_AL)**: AL methods that at each iteration greedily choose the least confident samples until the budget limitation is reached; **Binary Linear Programming Active Learning (BLP_AL)**: AL methods that at each iteration opt to minimize the sum of confidence scores of the chosen samples given the budget constraints. The optimization problem (see below) is solved using a BLP solver;[10] and **Binary Linear Programming (BLP)**: an algorithm that after training on the initial training set chooses all the samples at once, by solving the same constrained optimization problem as in BLP_AL.

For each of these categories, we experiment with four families of AL methods: **a) Unrestricted Disjoint Sets (UDJS):** This selection method is based on the non-aggregated multi-task confidence scores, where each sample can be chosen to be annotated on either task or both. The UDJS optimization problem aims to maximize the uncertainty scores $(1 - Conf_t(x))$ of the selected samples given the budget and selection constraints:

$$\begin{aligned}
\max \quad & \sum_{x\in U}\sum_{t\in T}(1 - Conf_t(x)) \cdot X_t(x) \\
\text{s.t.} \quad & \sum_{x\in U}\sum_{t\in T} Cost_t(x) \cdot (X_t(x) - Y(x)) \\
& \qquad + JCost(x) \cdot Y(x) \le B, \\
& \frac{1}{|T|}\sum_{t\in T} X_t(x) \ge Y(x) \quad \forall x \in U, \\
& X_t(x), Y(x) \in \{0,1\} \quad \forall x \in U, t \in T,
\end{aligned}$$

where U is the unlabeled set, T is the set of tasks, $Conf_t$ is the *MT-EC-t* confidence score, $X_t(x)$ is a binary indicator indicating the annotation of sample x on task t, and $Y(x)$ is a binary indicator indicating the annotation of x on all tasks.

Notice that this formulation may yield annotated examples for only one of the tasks, although this is unlikely, particularly under an iterative protocol when the confidence scores of the models are updated after each iteration.

b) Equal Budget Disjoint Sets (EQB-DJS): This strategy is similar to the above UDJS except that the budget is equally divided between the two tasks and the optimization problem is solved for each of them separately. If a sample is chosen to be annotated for both tasks, we update its cost according to the joint cost and re-solve the optimization problems until the entire budget is used.

c-f) Joint-task Selection: A sample could only be chosen to be annotated on both tasks, where confidence scores are calculated using a multi-task aggregation. The BLP optimization problem is formulated as follows:

$$\begin{aligned}
\max \quad & \sum_{x\in U}(1 - Conf(x)) \cdot Y(x) \\
\text{s.t.} \quad & \sum_{x\in U} JCost(x) \cdot Y(x) \le B, \\
& Y(x) \in \{0,1\} \quad \forall x \in U,
\end{aligned}$$

[10] https://www.python-mip.com/.

	ATIS				SNIPS			
	BERT		Roberta		BERT		Roberta	
	SF	ID	SF	ID	SF	ID	SF	ID
MT-MIN$_{BLP\_AL}$	84.53	**92.27**	**87.94**	**92.05**	**77.90**	**97.00**	**71.35**	95.29
MT-RRF$_{GRD\_AL}$	82.11	91.38	87.57	89.81	71.78	95.71	69.06	92.14
MT-AVG$_{BLP}$	**85.78**	89.92	86.38	90.03	73.46	96.57	69.73	**95.57**

Table 8: Comparison of decision categories (Q3.1 and Q3.2).

	ATIS				SNIPS			
	BERT		Roberta		BERT		Roberta	
	SF	ID	SF	ID	SF	ID	SF	ID
MT-MIN$_{BLP\_AL}$	84.53	**92.27**	**87.94**	**92.05**	**77.90**	**97.00**	**71.35**	**95.29**
UDJS$_{BLP\_AL}$	**87.46**	90.48	85.05	91.71	73.56	96.71	70.72	95.14
EQB-DJS$_{BLP\_AL}$	86.59	88.02	85.25	89.14	69.92	91.43	63.58	95.00

Table 9: Comparison of joint-task selection methods (Q3.3).

where $Conf$ is calculated by **c)** ***MT-AVG***, **d)** ***MT-MAX***, **e)** ***MT-MIN***, or **f)** ***MT-RRF***.[11]

g-j) Single-task Confidence Selection (STCS): A sample could only be chosen to be annotated on both tasks, where the selection process aims to maximize the uncertainty scores of only one of the tasks: **g) STCS-SF** or **j) STCS-ID**. Similarly to Joint-task Selection, the budget constraints are applied to the joint costs.

7.2 Research Questions

We focus on three research questions: **Q3.1**: Does BLP optimization improve upon greedy selection? **Q3.2**: Do optimization selection and active learning have a complementary effect? and **Q3.3**: Is it better to annotate all samples on both tasks or to construct a disjoint annotated training set?

7.3 Experimental Setup

We conduct experiments on two prominent datasets: ATIS (Price, 1990) and SNIPS (Coucke et al., 2018), and consider two Transformer-based encoders: BERT-base (Devlin et al., 2019) and Roberta-base (Liu et al., 2019b). Our code is largely based on the implementation of Zhu and Yu (2017).[12] We run the AL process for 5 iterations with an initial training set of 50 random samples and a fixed-size development set of 100 random samples. We train all models for 30 epochs per iteration, with an early stopping criterion and with label smoothing ($\alpha = 0.1$). Other hyper-parameters were set to their default values.

7.4 Results

Optimization-based Selection and AL (Q3.1 and Q3.2) To answer the first two questions, we show in Table 8 the final sample F1 performance for both tasks, when selection is done with the best selection method of each decision category: *MT-RRF*$_{GRD\_AL}$, *MT-MIN*$_{BLP\_AL}$, and *MT-AVG*$_{BLP}$. The results confirm that ***the BLP optimization (with AL) is indeed superior to greedy AL selection***, surpassing it in all setups (two tasks, two datasets, and two pre-trained language encoders, 5 of the comparisons are statistically significant).

The answer to Q3.2 is also mostly positive. BLP optimization and iterative AL have a complementary effect, as *MT-MIN*$_{BLP\_AL}$ in most cases achieves higher performance than *MT-AVG*$_{BLP}$ (50% of the results are statistically significant).

In fact, ***the answer for the two questions holds true for all selection methods***, as all perform best using our novel **BLP_AL** decision method. Second is **BLP**, indicating that the BLP formulation is highly effective for MT setups under different budget constraints. Finally, last is the standard greedy procedure (**GRD_AL**) which is commonly used in the AL literature.

Joint-task Selection (Q3.3) To answer Q3.3, we perform a similar comparison in Table 9, but now for the most prominent methods for each joint-task selection method. The results indicate that *MT-MIN*$_{BLP\_AL}$ that enforces ***joint annotation is better than allowing for non-restricted disjoint annotation*** (*UDJS*$_{BLP\_AL}$) ***and than equally splitting the budget between the two tasks*** (*EQB-DJS*$_{BLP\_AL}$), where 9 of the 16 comparisons are statistically significant. We hypothesize that the superiority of *MT-MIN*$_{BLP\_AL}$ stems from two main reasons: 1. Integrating the joint aggregation confidence score into the optimization function provides a better signal than the single-task confidence scores; 2. Having a joint dataset where all samples are annotated on both tasks rather than a disjointly annotated (and larger) dataset allows the model to achieve better performance since the two tasks are closely related.

[11]Since *MT-PAR* and *MT-IND* do not score the examples, they can only be applied with the greedy decision method. A discussion on their results is provided in §8.

[12]`https://github.com/sz128/slot_filling_and_intent_detection_of_SLU`.

	DP + NER		NER + RE		SF + ID		
	DP	NER	NER	RE	SF	ID	Average
MT-R	86.40	70.78	81.96	52.47	76.94	91.95	76.75
MT-AVG	**89.03**	74.44	80.23	55.97	76.42	91.09	77.86
MT-MAX	88.64	**75.90**	81.50	**57.74**	76.64	89.17	78.27
MT-MIN	88.73	74.77	81.32	54.76	73.75	91.74	77.51
MT-PAR	88.31	75.86	81.47	56.92	**77.72**	90.28	**78.43**
MT-RRF	88.31	75.89	81.78	54.29	77.63	**92.26**	78.36
MT-IND	87.54	74.83	**82.13**	54.85	76.67	91.44	77.91

Table 10: Average results of the joint multi-task selection methods across all three setups.

Finally, we also report that ST-AL experiments have led to poor performance. The ST-AL methods trail by 8.8 (SF) and 4.7 (ID) F1 points from the best MT-AL method *MT-MIN*$_{BLP\_AL}$ on average. Interestingly, we also observe that selection according to ID (*STCS-ID*$_{BLP\_AL}$) has led to better average results on both tasks than selection according to SF (*STCS-SF*$_{BLP\_AL}$), suggesting that similarly to §6, selection according to the higher-level task often yields better results.

8 Overall Comparison

As a final evaluation, we turn to compare the performance of our proposed joint-selection MT-AL methods in all setups. In our first setup (§5) we implemented all MT-AL selection methods with greedy selection, considering uniform task importance. Thereafter (§6 and §7), we showed how these selection methods can be modified in the next two setups by either integrating non-uniform task weights or replacing the greedy selection with a BLP loss function overconfidence scores. However, not all of our MT-AL selection methods can be modified in such ways.[13] We hence report our final comparison given the conditions applied in the first setup: Assuming uniform task weights and selecting samples in a greedy manner. Table 10 summarizes the average performance of the MT-AL methods in our three proposed setups: Complementing tasks (DP + NER), hierarchically related tasks (NER + RE) and tasks of different annotation granularity (SF + ID).

Each selection method has its cons and pros, which we outline in our final discussion.

[13]See footnotes 7 and 11 for further details.

MT-R, not surprisingly, is the worst performing method on average, as it makes no use of the model's predictions. Nevertheless, the method performs quite well on the third setup (SF + ID) when compared to the other methods that were trained with the greedy decision method, the least successful decision method of this setup. Next, ***MT-AVG*** performs well when the tasks are of equal importance (DP + NER), but achieves only moderate performance on the other setups.

Surprisingly, ***MT-MAX*** is highly effective despite its simplicity. It is mostly beneficial for the first two setups (DP + NER and NER + RE), where the tasks are of the same annotation granularity. It is the third best method overall, and it does not lag substantially behind the best method, *MT-PAR*. Interestingly, ***MT-MIN***, which offers a complementary perspective to *MT-MAX*, is on average the worst MT-AL method, excluding *MT-R*, and is mainly beneficial for the first setup (DP + NER).

The next MT-AL method, ***MT-PAR***, seems to capture well the joint confidence space of the task pairs. It is the best method on average, achieving high average scores in all setups. However, when incorporating it with other training techniques, such as applying non-uniform weights (for the second setup), it is outperformed by the other MT-AL methods. ***MT-RRF*** does not lag far behind *MT-PAR*, achieving similar results on most tasks, excluding the RE and ID tasks, which are the higher-level tasks of their setups. Finally, ***MT-IND*** does not excel in three of the four tasks of the first two setups, while achieving the best average results on NER, when jointly trained with RE. Furthermore, the method demonstrates strong performance on the third setup, when the tasks are of different annotation granularity, justifying the independent annotation selection in this case.

9 Conclusions

We considered the problem of multi-task active learning for pre-trained Transformer-based models. We posed multiple research questions concerning the impact of multi-task modeling, multi-task selection criteria, overconfidence reduction, the relationships between the participating tasks, as well as budget constraints, and presented a systematic algorithmic and experimental investigation in order to answer these questions. Our results demonstrate the importance of MT-AL modeling in three challenging real-life scenarios, corresponding to diverse relations between the participating tasks. In future work, we plan to research setups with more than two tasks and consider language generation and multilingual modeling.

Acknowledgments

We would like to thank the action editor and the reviewers, as well as the members of the IE@Technion NLP group for their valuable feedback and advice. This research was partially funded by an ISF personal grant no. 1625/18. R. R. also acknowledges the support of the Schmidt Career Advancement Chair in AI.

References

Guirong Bai, Shizhu He, Kang Liu, Jun Zhao, and Zaiqing Nie. 2020. Pre-trained language model based active learning for sentence matching. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1495–1504.

Ying Chen, Wenjun Hou, Xiyao Cheng, and Shoushan Li. 2018. Joint learning for emotion classification and emotion cause detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 646–651, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1066

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759. https://doi.org/10.1145/1571941.1572114

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: An embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302. https://doi.org/10.18653/v1/2020.emnlp-main.21

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.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392. https://doi.org/10.18653/v1/P18-1128

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2018. Active learning for deep semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 43–48, Melbourne, Australia. Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-2008

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active learning for BERT: An empirical study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics. `https://doi.org/10.18653/v1/2020.emnlp-main.638`

Jenny Rose Finkel and Christopher D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334. `https://doi.org/10.3115/1620754.1620802`

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics. `https://doi.org/10.18653/v1/W17-3518`

Daniel Grießhaber, Johannes Maucher, and Ngoc Thang Vu. 2020. Fine-tuning BERT for low-resource natural language understanding via active learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1158–1171, Barcelona, Spain (Online). International Committee on Computational Linguistics. `https://doi.org/10.18653/v1/2020.coling-main.100`

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 415–423. `https://doi.org/10.3115/1620754.1620815`

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780. `https://doi.org/10.1162/neco.1997.9.8.1735`, PubMed: 9377276

Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60. `https://doi.org/10.3115/1614049.1614064`

Fariz Ikhwantri, Samuel Louvan, Kemal Kurniawan, Bagas Abisena, Valdi Rachman, Alfan Farizki Wicaksono, and Rahmad Mahendra. 2018. Multi-task active learning for neural semantic role labeling on low resource conversational corpus. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 43–50. `https://doi.org/10.18653/v1/W18-3406`

Seokhwan Kim, Yu Song, Kyungduk Kim, Jeong-Won Cha, and Gary Geunbae Lee. 2006. Mmr-based active machine learning for bio named entity recognition. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 69–72.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*. `https://arxiv.org/abs/1412.6980`

Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, and Chao Zhang. 2020. Calibrated language model fine-tuning for in- and out-of-distribution data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1326–1340. Online. Association for Computational Linguistics. `https://doi.org/10.18653/v1/2020.emnlp-main.102`

Chaitanya Kulkarni, Wei Xu, Alan Ritter, and Raghu Machiraju. 2018. An annotated corpus for machine reading of instructions in wet lab protocols. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 97–106, New Orleans, Louisiana. Association for Computational Linguistics. https://doi.org/10.18653/v1/N18-2016

Belinda Z. Li, Gabriel Stanovsky, and Luke Zettlemoyer. 2020. Active learning for coreference resolution using discrete annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8320–8331, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.738

Jiacheng Li, Haibo Ding, Jingbo Shang, Julian McAuley, and Zhe Feng. 2021. Weakly supervised named entity tagging with learnable logical rules. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4568–4581, Online. Association for Computational Linguistics.

Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. 2016. Active learning for dependency parsing with partial annotation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 344–354.

Ying Lin, Shengqi Yang, Veselin Stoyanov, and Heng Ji. 2018. A multi-lingual multi-task architecture for low-resource sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 799–809. https://doi.org/10.18653/v1/P18-1074

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Alexander V. Lotov and Kaisa Miettinen. 2008. Visualizing the Pareto frontier. In *Multiobjective Optimization*, pages 213–243. Springer. https://doi.org/10.1007/978-3-540-88908-3_9

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1360

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6297–6308.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8528–8535. https://doi.org/10.1609/aaai.v34i05.6374

Linh The Nguyen and Dat Quoc Nguyen. 2021. PhoNLP: A joint multi-task learning model for Vietnamese part-of-speech tagging, named entity recognition and dependency parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 1–7, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-demos.1

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal

Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 32:13991–14002.

Álvaro Peris and Francisco Casacuberta. 2018. Active learning for interactive neural machine translation of data streams. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 151–160, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/K18-1015

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. https://doi.org/10.18653/v1/N18-1202

Patti Price. 1990. Evaluation of spoken language systems: The atis domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*. https://doi.org/10.3115/116580.116612

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Roi Reichart and Ari Rappoport. 2007. An ensemble method for selection of high quality parses. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 408–415.

Roi Reichart and Ari Rappoport. 2009. Sample selection for statistical parsers: Cognitively driven algorithms and evaluation measures. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 3–11. https://doi.org/10.3115/1596374.1596379

Roi Reichart, Katrin Tomanek, Udo Hahn, and Ari Rappoport. 2008. Multi-task active learning for linguistic annotations. In *Proceedings of ACL-08: HLT*, pages 861–869.

Guy Rotman and Roi Reichart. 2019. Deep contextualized self-training for low resource dependency parsing. *Transactions of the Association for Computational Linguistics*, 7:695–713. https://doi.org/10.1162/tacl_a_00294

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

Niloofar Safi Samghabadi, Parth Patwa, Srinivas PYKL, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and misogyny detection using BERT: A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131, Marseille, France. European Language Resources Association (ELRA).

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6949–6956. https://doi.org/10.1609/aaai.v33i01.33016949

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. https://doi.org/10.18653/v1/P18-1018

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079. https://doi.org/10.3115/1613715.1613855

H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. 1992. Query by committee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 287–294.

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 252–256, Vancouver, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/W17-2630

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235. https://doi.org/10.18653/v1/P16-2038

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826. https://doi.org/10.1109/CVPR.2016.308

Sunil Thulasidasan, Gopinath Chennupati, Jeff A. Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. 2019. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. https://doi.org/10.2172/1525811

Katrin Tomanek and Udo Hahn. 2010. A comparison of models for cost-sensitive active learning. In *Coling 2010: Posters*, pages 1247–1255.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-5446

Maciej Wiatrak and Juha Iso-Sipila. 2020. Simple hierarchical multi-task neural end-to-end entity linking for biomedical text. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 12–17, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.louhi-1.2

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. 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. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Kaige Xie, Cheng Chang, Liliang Ren, Lu Chen, and Kai Yu. 2018. Cost-sensitive active learning for dialogue state tracking. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 209–213, Melbourne, Australia. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-5022

He Zhao, Longtao Huang, Rong Zhang, Quan Lu, and Hui Xue. 2020. SpanMlt: A span-based multi-task learning framework for pair-wise aspect and opinion terms extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3239–3248, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.296

Hua Zhu, Wu Ye, Sihan Luo, and Xidong Zhang. 2020. A multitask active learning framework for natural language understanding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4900–4914, Barcelona, Spain (Online). International Committee on Computational Linguistics. `https://doi.org/10.18653/v1/2020.coling-main.430`

Su Zhu and Kai Yu. 2017. Encoder-decoder with focus-mechanism for sequence labelling based spoken language understanding. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5675–5679. IEEE. `https://doi.org/10.1109/ICASSP.2017.7953243`