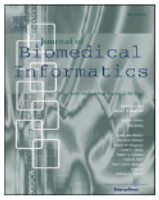
Journal of Biomedical Informatics 130 (2022) 104062

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin



Effects of data and entity ablation on multitask learning models for biomedical entity recognition

Nicholas E. Rodriguez [a], Mai Nguyen [b], Bridget T. McInnes [a,*]

[a] Department of Computer Science, Virginia Commonwealth University, Richmond 23284, USA
[b] San Diego Supercomputer Center, University of California, San Diego 92093, USA

ARTICLE INFO

Keywords:
Natural language processing
Named entity recognition
Deep learning
Machine learning
Biomedical text processing

ABSTRACT

Motivation: Training domain-specific named entity recognition (NER) models requires high quality hand curated gold standard datasets which are time-consuming and expensive to create. Furthermore, the storage and memory required to deploy NLP models can be prohibitive when the number of tasks is large. In this work, we explore utilizing multi-task learning to reduce the amount of training data needed to train new domain-specific models. We evaluate our system across 22 distinct biomedical NER datasets and evaluate the extent to which transfer learning helps task performance using two forms of ablation.

Results: We found that multitasking models generally do not improve performance, but in many cases perform on par compared to single-task models. However, we show that in some cases, new unseen tasks can be trained as a single model using less data by starting with weights from a multitask model and improve performance.

Availability: The software underlying this article are available in: https://github.com/NLPatVCU/multitasking_bert-1.

1. Introduction

Named entity recognition (NER) is a highly utilized task in Natural Language Processing (NLP) and involves labelling a sequence of words or tokens with their appropriate entity labels. For example, given a sentence (shown in Fig. 1), identify that *Streptococcus lividans* is a Species and *A21978C* is a Gene. It can be used to identify entities of interest within documents and is often one of the first tasks performed on documents before downstream text mining tasks are performed [1]. Consequently, errors in NER models can be propagated to downstream tasks affecting the overall performance of text processing pipelines.

Training domain-specific NER models requires the creation of high quality gold standard datasets that are hand-curated by domain experts. This annotation process is time-consuming and expensive. Therefore, in addition to improving performance of NER models, we are especially interested in reducing the amount of training data needed to train new domain-specific models. If two or more tasks' target entities are similar, like chemicals, genes, and proteins, then training millions of parameters per model may be redundant in certain use cases, especially if a multitasking model provides comparable performance. Previous work shows that pairing similar tasks [2], e.g. by similar topics, can improve performance of multitasking models [3]. We seek to understand if there is mutually beneficial information with respect to task performance that the base encoder is using to generate contextualized embeddings that contribute to gains from multitasking learning. Although, that is difficult to measure directly. In this work, we evaluate the effectiveness of training multiple neural network-based models on various biomedical NER tasks using two forms of ablation.

1. Data ablation, which we use to assess how much data is needed to effectively train each model.
2. Entity ablation, which we use to assess how well these models can be updated with a new task after they've been multi-task trained.

Our results show:

1. The extent to which multitasking model performances compare to those of single task models using a fraction of the training data and only one set of embedding-producing transformer encoders.
2. The extent to which transformer-based multitasking models scale to over 20 biomedical datasets and tasks.

* Corresponding author.
E-mail address: btmcinnes@vcu.edu (B.T. McInnes).

https://doi.org/10.1016/j.jbi.2022.104062
Received 23 August 2021; Received in revised form 11 February 2022; Accepted 27 March 2022
Available online 9 April 2022

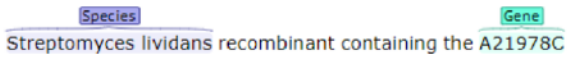


Fig. 1. Named entity recognition example.

3. A use-case for this method by showing that new unseen tasks can be learned by a single model using less data by starting with weights from a multitask model.

2. Background

2.1. Transformer encoders

Self-attention mechanisms [4] have been shown to outperform previous state-of-the-art algorithms for sequence learning including bidirectional long short-term memory (biLSTM) units, conditional random fields (CRFs), and other recurrent neural networks (RNNs). One major benefit to self-attention-based models is their ability to create contextualized token embeddings and interpretable alignment matrices. Devlin et al. [5] proposed bi-directional encoder representations from transformers (BERT), which uses twelve sequential transformer layers (base version) and a simple feed-forward neural network (FFNN) to perform language modelling tasks on unlabelled data. Those tasks include next-sentence prediction and masked token prediction. The result of this language modelling approach is a word/token-encoder that produces contextualized token embeddings that have the capacity to represent a range of useful information that would otherwise require extensive feature engineering and selection (a contextual embedding for a token depends on the context in which the token appears in a sentence). More importantly to our objective here, in addition to providing contextual embeddings, BERT models can be finetuned such that the resulting embeddings are also *domain-specific*. BioBERT leveraged this feature and Lee et al. [6] adapted the pretraining data to include biomedical texts. In this work, the transformer encoders are initialized from BioBERT model weights and vocabulary.

2.2. Named entity recognition

The goal of NER is to extract information from text about the location of entities of interest. Here, we define this task as a sequence labelling problem, where given an input sequence of tokens, in this case, subword tokens, a classification model predicts a sequence of corresponding labels. Each classifier predicts an entity type for a single dataset.

Note that a dataset can be annotated with multiple entity types, and in that case we train a separate classifier for each of the different entity types. Although it is possible to combine datasets containing annotations of the same entity type (resulting in fewer models), we chose to train the models separately to measure the performance of the models on datasets with different sizes, as well as token and annotation distributions.

Subword classification models used here are neural networks and contain two main components: an embedding layer and a task-specific classification layer. The classification layer is a fully connected feed-forward network followed by a softmax layer which computes the probability distribution over all possible labels. The embedding layer is a stack of transformer encoders that outputs a sequence of contextual embeddings which are then passed to the classification layer one at a time. For the embedding layer, we use BioBERT-Base v1.1 [6].

3. Method

Two model architectures are used in this work. The first is a single task learning architecture which contains a transformer-based embedding layer and a linear layer. The other is a multitask learning architecture with one linear layer per task and one embedding layer that is shared between them.

3.1. Single task learning

Fig. 2 shows the architecture of the single task model, i.e., each dataset is used to train a separate model. Each single task learning model contains one transformer-based embedding layer per task-specific layer, as shown in Fig. 2. Given an input sequence of subword token encodings $\boldsymbol{s} = (s_1, s_2, \ldots, s_n)$, the embedding layer, indexed by i, produces a sequence of continuous representations (embeddings) $\boldsymbol{e_i} = (e_{i1}, e_{i2}, \ldots, e_{in})$ from which the i-th task-specific layer can assign a sequence of labels $\boldsymbol{l_i} = (l_{i1}, l_{i2}, \ldots, l_{in})$. The input sequence and output of the embedding layer have the same dimensions. Since this work focuses on named entity recognition, the output of the task-specific layer will have the same dimensions as the input sequence $\boldsymbol{s}$ and each embedding layer output $\boldsymbol{e_i}$.

3.2. Multitask learning

The embedding and task-specific layers of the multitask models are identical to their single task model counterparts with one exception. Instead of using one embedding layer per task-specific layer, all of the task-specific layers receive their input from the same embedding layer. Fig. 3 shows the architecture of the multitask model, where the encoder layer is shared between all tasks. There is no weight sharing or representation sharing between task-specific layers other than the encoder. Given $\boldsymbol{s}$, the single embedding layer produces an embedding sequence $\boldsymbol{e} = (e_1, e_2, \ldots, e_n)$ from which the i-th task-specific layer can generate a sequence of labels $\boldsymbol{l_i} = (l_{i1}, l_{i2}, \ldots, l_{in})$.

For each epoch and dataset, a batch of examples from the dataset are passed to the embedding layer, then to its respective subword classification layer. Back-propagation occurs before continuing with the next dataset. A single epoch is complete when all training examples from all datasets have passed through the model one time.

We initially used the same round robin procedure used by Mulyar et al. [7]; that is if different datasets have an unequal number of batches, then batches from the smaller datasets are resampled until the sampling of all batches in the largest dataset is complete (this is also called upsampling). Overfitting can occur as a result of this type of sampling approach, however, in practice we only found one example of severe overfitting, and interestingly, other than that one example, upsampling had no effect on model performance. Ultimately, we chose to eliminate upsampling in favor of faster training.

4. Evaluation methodology

The performance of all models is evaluated using precision, recall, and F_1 score. As per [8] metrics are calculated at the IOB tag-level or the entity mention-level. IOB tag evaluation is performed at the word-level by selecting the label corresponding to the last subword token in a word.

Training NER models using subword embeddings as opposed to word-level embeddings introduces additional factors affecting loss calculation and evaluation. There are various strategies for dealing with

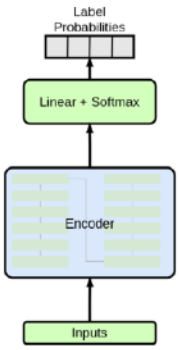


Fig. 2. Overview of our single task architecture.

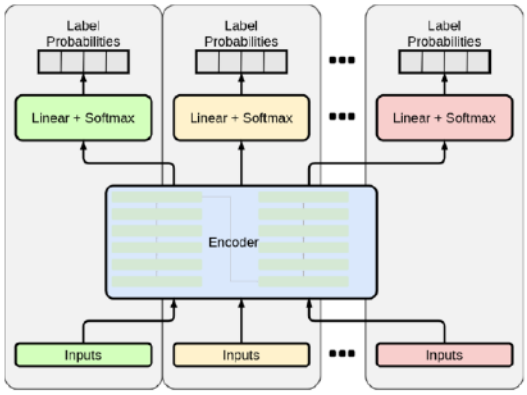


Fig. 3. Overview of our multitask architecture.

subword embeddings, such as computing a word-level embedding from the average of its subword embeddings, or simply considering only the first or last subword embedding within a word [9]. These strategies may be useful when comparing models trained from different algorithms that do not use subword tokenization. One advantage to using all subword embeddings for loss calculation is that models can be flexibly reused for inference in real-world applications by allowing the whole word or parts of the word to be labelled. It also allows for different post-processing methods to be compared without biasing the predictions by the training strategy. A potential drawback to subword independence, however, is that when used for IOB tag-level evaluation it can create more opportunities for mislabelling and lead to underestimation of performance compared to word-level evaluation. However, we find that both IOB tag evaluation at the word and subword token-level tend to overestimate performance compared to entity mention-level evaluation.

Because training examples are batched, all input sequences have the same fixed length and may contain padding tokens at the end of the sequence if the number of subword tokens is less than the fixed input length. Logits at positions corresponding to padding tokens as well as other special BERT tokens are removed, and a label is assigned to each subword.

The output of the classifiers is a vector with its length equal to the number of labels. For IOB-labelled datasets, the output vector for each subword has a length of 4, including one for BERT-TOKEN. Applying a softmax operation to that output vector gives the probability distribution over all labels, and taking the argmax of that vector gives the predicted label for a subword. If a word has multiple subwords, the word is assigned the label of the last subword. All predicted word-level labels in a dataset are concatenated and used to calculated precision, recall, and F_1 score. Assuming the contribution of each dataset metric is equal regardless of size, the class average for each entity type is computed as the average of the individual dataset scores within that class. Correlation coefficients, r^2, and p-values between $F1$ scores and dataset features are computed using linear regression and Pearson's r.

4.1. Data ablation

Two of the main objectives in this work is to understand how much training data is needed to effectively train multitasking models for NER and how much training data is needed to update a multitask-learned model for a new entity type. This was done using two approaches: (1) data ablation and (2) entity ablation.

We used *data ablation* to assess the required amount of training data needed to train a multitask model. Data ablation in this case means that a proportion of training examples from each dataset were removed before training. For example, an ablation amount of 0.9 means that 90% of the training data was removed before training the model, and an ablation amount of 0 means that none of other training data was removed before training the model. The number of examples in the development and test sets remained the same.

We use *entity ablation* to assess the amount of training data require to update a multitask for a new entity type. We refer to entity ablation to mean that during the initial training of a multitask model, all datasets containing annotations of a given entity type are removed. Then, the weights from the shared embedding layer are used to initialize a shared new embedding layer, and the previously ablated datasets are subsequently used for training new subword classification layers in a single task learning model.

5. Data

In this work, we utilize 22 datasets from the biological domain. Each dataset was annotated with one or more of the entity classes *cell line*, *chemical*, *gene*, and *species*. Table 1 shows the total and unique counts of the mentions in each dataset; where a mention is an instance of an entity. The *Type* column of Table 1 indicates the document type, where the first letter indicates ***Article*** or ***Patent***, and the second letter indicates ***Abstract*** or ***Full-text***.

For model training and evaluation, the datasets were divided at the document level into training, testing, and development sets with proportions equal to 0.6, 0.3, and 0.1, respectively. We use the same partitioning strategy as described by Weber et al. [10].

The data underlying this article was accessed using HUNER at https://github.com/hu-ner/huner.

6. Experimental details

All datasets were first preprocessed according to Weber et al. [10]. The document text was tokenized using OpenNLP [11]. Entity labels were tagged using the IOB scheme, which indicates whether a token is inside, outside, or at the beginning of an entity mention. WordPiece tokenization [5] was performed on each token using a cased vocabulary. We refer to the resulting atomic elements as subword tokens.

The transformer embedding layers used here are constrained to a maximum input sequence length of 512. In order to maximize the context available to the encoder layers, the input sequence length of subword tokens were kept as close to 512 as possible, while also preventing a mention from being split at the end of the sequence. In the case that a mention is located at the end of an input sequence, the sequence is truncated to exclude the mention, and the mention is moved to the subsequent input sequence. Supplemental Fig. 1 shows the counts of input sequences after preprocessing for each training set. The smallest training set is CLL cell line with 14 input sequences. The largest is CHEMDNER with 4182 input sequences in its training set.

Supplemental Table 1 shows the average sequence counts (training sets only) for each entity type. Species datasets have an average of 263 sequences, the smallest of the entity types. The largest is chemical, with an average of 1619 sequences.

Various pre-processing methods can have an impact on model performance. For example, we observed a slight increase in performance when using cased tokenization, maximizing the context window of input sequences (instead of sentence segmentation), and using pre-tokenized input to the WordPiece tokenizer.

7. Results and discussion

In this section, we present and discuss our NER results over the 22 datasets, which were annotated with entities *cell line*, *chemical*, *gene*, and *species* in IOB format. We compare two types of models: single task and multitask. Additionally, data ablation and entity ablation were performed. Our objective is to capture mutually beneficial information into a single jointly-learned model and minimize the amount of training data required to train new unseen tasks. Here we present the results of single task, multitask, and ablation experiments.

Table 1
The mention counts over the development, test and training data for each dataset.

			Total			Unique		
Entity	Dataset	Type	DEV	TEST	TRAIN	DEV	TEST	TRAIN
CELL LINE	CLL	AA	30	77	234	26	67	195
	GELLUS	AA	75	247	328	32	99	110
	JNLPBA	AA	429	1117	2284	286	771	1383
CHEMICAL	CDR	AA	1511	4716	9207	560	1503	2461
	CEMP	PF	6364	18958	39293	3093	7506	14240
	CHEBI	PF	1262	6067	8779	594	2077	2627
	CHEMDNER	AA	8062	24288	48347	3687	9035	16094
	SCAI	AA	83	375	852	59	252	521
GENE	BC2GM	AA	2163	6753	14456	1938	5513	10846
	BIOINFER	AA	455	1383	2658	244	597	987
	DECA	AA	576	1776	3670	250	772	1457
	FSU	AA	6606	19383	33505	2539	6429	9878
	GPRO	PA	1315	3576	7832	900	2004	3958
	IEPA	AA	104	300	708	46	81	146
	JNLPBA	AA	3029	8777	18463	1306	3195	6029
	MIRNA	AA	76	291	541	38	129	234
	OSIRIS	AA	96	291	535	34	114	234
	VARIOME	AF	300	1082	3045	65	214	509
SPECIES	LINNEAUS	AF	85	278	566	26	41	70
	MIRNA	AA	64	227	385	12	34	31
	S800	AA	406	1074	2188	203	518	1044
	VARIOME	AF	33	66	83	3	7	6

7.1. Single task learning results

In this section, we describe our single task learning baseline results. Table 2 shows the results of the single task learning experiments. The datasets are grouped by entity type. The development sets containing 10% of examples were used to calculate F_1 score, precision, and recall during model training. After 20 epochs of training, the model with the highest F_1 score when evaluated on the development set was selected for

Table 2
Single task learning results.

			F_1		Precision		Recall	
Entity	Dataset	Epoch	DEV	TEST	DEV	TEST	DEV	TEST
CELL LINE	CLL	19	0.857	0.885	0.818	0.830	0.900	0.948
	GELLUS	10	0.943	0.864	0.943	0.971	0.943	0.778
	JNLPBA	9	0.830	0.808	0.802	0.847	0.860	0.772
	average		0.877	0.852	0.854	0.883	0.901	0.833
CHEMICAL	CDR	11	0.947	0.934	0.932	0.925	0.963	0.944
	CEMP	1	0.919	0.913	0.884	0.885	0.957	0.943
	CHEBI	15	0.905	0.894	0.886	0.916	0.924	0.874
	CHEMDNER	6	0.944	0.944	0.932	0.951	0.957	0.936
	SCAI	14	0.970	0.939	0.977	0.928	0.964	0.950
	average		0.937	0.925	0.922	0.921	0.953	0.929
GENE	BC2GM	15	0.903	0.905	0.889	0.904	0.918	0.906
	BIOINFER	19	0.945	0.929	0.933	0.929	0.958	0.928
	DECA	16	0.698	0.711	0.664	0.702	0.736	0.720
	FSU	19	0.937	0.943	0.917	0.927	0.958	0.961
	GPRO	6	0.817	0.788	0.739	0.752	0.914	0.828
	IEPA	13	0.872	0.907	0.868	0.905	0.875	0.910
	JNLPBA	7	0.908	0.902	0.889	0.874	0.928	0.931
	MIRNA	13	0.857	0.747	0.842	0.708	0.873	0.790
	OSIRIS	18	0.878	0.812	0.867	0.897	0.889	0.741
	VARIOME	19	0.953	0.924	0.930	0.941	0.978	0.907
	average		0.877	0.857	0.854	0.854	0.903	0.862
SPECIES	LINNEAUS	16	0.857	0.771	0.853	0.963	0.862	0.643
	MIRNA	11	0.985	0.906	0.985	0.990	0.985	0.835
	S800	15	0.873	0.831	0.877	0.827	0.869	0.836
	VARIOME	19	0.618	0.548	0.773	0.586	0.515	0.515
	average		0.833	0.764	0.872	0.842	0.808	0.707
OVERALL AVERAGE			0.883	0.855	0.873	0.871	0.897	0.845

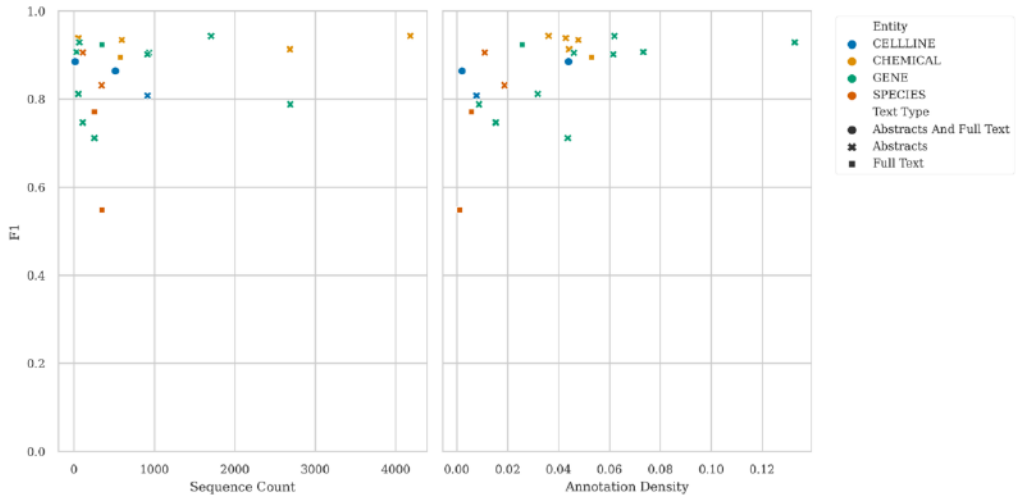


Fig. 4. Relationships between sequence counts and annotation density on single task learning performance (F_1 score).

prediction and evaluation on the corresponding test set (30% of examples). For each task (dataset-entity pair), the enumerated epoch value corresponding to the highest F_1 score on the development set is shown in the *Epoch* column of Table 2. F_1 *score*, *Precision*, and *Recall* are shown in subsequent columns. Summary statistics are given below each entity type. Average F_1 scores (test set) are 0.852, 0.925, 0.857, and 0.764 for cell line, chemicals, gene, and species, respectively. These single task learning results serve as a baseline for subsequent multitask learning and ablation experiments described in the following sections.

Models predicting chemical entities have the highest F_1 scores with an average of 0.925 on test sets. Species models seem to have the lowest F_1 scores. The VARIOME species model is the lowest performing model in this entity group. This may be explained in part by the fact that it contains six unique mentions (Table 1) in the training set and only *B* and *O* annotations, and thus the model has less context from which to learn information about complete spans corresponding to mentions containing more than one token or subword token.

Similarly, the MIRNA species model has a F_1 score of 0.906, the highest F_1 score of the species models, which is not surprising considering the lexical diversity of mentions is quite limited. There are 31 unique mentions including *patient*, *rat*, *human*, and other mentions that a biologist might consider to be common organism names rather than species given in binomial nomenclature, which is a more difficult task for an NER model because it requires information about a potential mention's morphology and the meaning of special characters, e.g. a period to abbreviate genus as in *E. coli*.

Fig. 4 shows the relationship between the performance of single task learning (STL) models and the distribution of their sequence count and annotation density within their respective training sets. Annotation density is defined here as ratio of mentions to tokens. We observe a moderate correlation between F_1 scores and the annotation density ($r = 0.52, r^2 = 0.27, p = 0.01$). From this we conclude that there are likely other factors contributing to the variation in performance besides the density of mentions within the documents. We found no other significant relationships between dataset features such as sequence count, token count, and document type.

7.2. Multitask learning results

In this section we describe our multitask learning results. Here, we present the results when training a multitask model on all 22 tasks and report the F_1 score, precision and recall. We used the same hyperparameters for the multitasking models as for the single task models; as our objective is not to find the best possible models, but rather to provide a baseline for MTL for comparison to STL and ablation results. Additionally, we demonstrate the extent to which this approach can be leveraged to reduce the number of embedding-producing model layers needed to perform NER tasks on many datasets with different entity classes.

Table 3 shows the results of the multitask learning experiment. All 22 NER tasks (dataset-entity pairs) were trained using a single, shared encoder which provides contextual embeddings to each subword classification layer. F_1 score, precision, and recall scores are reported. Additionally, these scores are summarized below each entity group.

Two of the five chemical models performed slightly better when trained in a multitask setting. Three gene models outperformed their STL counterparts. MIRNA gene showed the greatest improvement in performance within the gene group (+0.048). Of the species group, VARIOME was the only model that showed improvement over STL which had the largest increase in performance of all models +0.141. Multitask learning hurt the performance of all cell line models. The difference in STL and MTL F_1 score averages for cell line is −0.065. However, the difference between the overall STL and MTL performance for any of the datasets is not statistically significant.

We conclude that overall, multitasking at this scale with these datasets does not generally lead to an overall improvement in performance over single task learning. The models seem to compete with each other, as some F_1 scores are volatile during training compared to STL, and do not converge before 20 epochs. However, it may be possible to tune the models such that each classification layers converges more quickly, such tuning is outside of the scope of this work, and is investigated in future work.

7.3. Data ablation results

In this section we present the results of the data ablation experiments. We trained the multitasking and single task models on progressively smaller training sets while keeping the development and test sets the same as their original size. By removing, or ablating, data training examples, we can observe the performance of the both types of models given limited training data.

The results in Tables 4 and 5 show the performances of STL and MTL models with data ablation. We have included the average STL results in Table 5 for each entity type to aid in the comparison. The *Ablation amount* column indicates the proportion of training data removed from each dataset. For example, the column *0.90* shows the performance of models when trained on 10% of their training sets, and *0.00* shows the

Table 3
Multitask learning results.

		F_1			Precision		Recall	
Entity	Dataset	DEV	TEST	STL	DEV	TEST	DEV	TEST
CELL LINE	CLL	0.825	0.795	0.885	0.788	0.742	0.867	0.857
	GELLUS	0.939	0.824	0.864	0.942	0.941	0.936	0.732
	JNLPBA	0.827	0.743	0.808	0.822	0.704	0.833	0.786
	average	0.864	0.787	0.852	0.851	0.796	0.878	0.792
CHEMICAL	CDR	0.959	0.947	0.934	0.964	0.946	0.954	0.947
	CEMP	0.920	0.915	0.913	0.882	0.892	0.962	0.939
	CHEBI	0.897	0.888	0.894	0.878	0.908	0.916	0.868
	CHEMDNER	0.942	0.941	0.944	0.934	0.946	0.950	0.937
	SCAI	0.969	0.826	0.939	0.964	0.763	0.974	0.902
	average	0.937	0.903	0.925	0.924	0.891	0.951	0.919
GENE	BC2GM	0.885	0.875	0.905	0.879	0.915	0.892	0.838
	BIOINFER	0.911	0.920	0.929	0.925	0.925	0.898	0.915
	DECA	0.685	0.708	0.711	0.645	0.686	0.730	0.730
	FSU	0.932	0.926	0.943	0.923	0.922	0.940	0.930
	GPRO	0.813	0.800	0.788	0.740	0.733	0.902	0.882
	IEPA	0.856	0.809	0.907	0.811	0.863	0.906	0.762
	JNLPBA	0.901	0.881	0.902	0.897	0.882	0.906	0.881
	MIRNA	0.856	0.795	0.747	0.848	0.740	0.864	0.859
	OSIRIS	0.953	0.837	0.812	0.949	0.879	0.957	0.799
	VARIOME	0.943	0.918	0.924	0.912	0.906	0.976	0.931
	average	0.874	0.847	0.857	0.853	0.845	0.897	0.853
SPECIES	LINNEAUS	0.856	0.721	0.771	0.860	0.606	0.851	0.890
	MIRNA	0.840	0.825	0.906	0.926	0.902	0.769	0.760
	S800	0.852	0.799	0.831	0.825	0.774	0.881	0.825
	VARIOME	0.765	0.689	0.548	0.743	0.622	0.788	0.773
	average	0.828	0.759	0.764	0.838	0.726	0.822	0.812
OVERALL AVERAGE		0.879	0.836	0.855	0.866	0.827	0.893	0.852

performance of the models when trained on their full training sets [1].

When comparing average F_1 scores within entity groups, the MTL chemical models have a negligible improvement in performance over STL at ablation amounts 0.50, 0.75, and 0.90. The MTL gene models averages show a modest improvement at ablation amounts 0.75 and 0.90. However, the cell line and species models vary more compared to chemical and gene models. At 0.90 ablation, all species models considerably decrease in average performance regardless of the training type, but remain on par at lower ablation amounts. The cell line MTL model's average performance is lower than its STL counterpart, until 0.90 ablation.

Overall the results indicate that the training data amount is decreased the performance of the single task learning degrades at a higher rate than the multitask environment.

7.4. Entity ablation results

A primary goal of this work is to understand how much data is required to update a multitask model for a new unseen task. This simulates a real-world scenario where domain experts annotate a collection of documents with entities, then iteratively add more useful entities and features for extraction from the documents. In this section, we explore interesting features of the multitask models and evaluate a use case for when a new unseen entity type is trained on limited data using previously multitask-trained models.

In this experiment, MTL models first underwent a round of finetuning which included all datasets *except* those from one entity type. Then, a second round of finetuning was performed on each of the previously excluded datasets independently. By removing one entity group from MT-training in the first round of finetuning, we should be able to observe whether there are interactions between classes that can affect performance of tasks in the multitasking setting. Using these entity-ablated MTL models, we continue to finetune them on a single task to determine if any tasks can benefit from MTL pre-training.

Fig. 5 shows the result of the first round of finetuning. The bottom axis shows the task group and the bars show the F_1 score distribution of the task group in the absence of a group indicated by color. For example, the first bar (blue) on the left shows the F_1 scores of chemical tasks after multitask training with all tasks *except* cell line tasks. The results show that the cell line and species tasks are most dramatically impacted by entity ablation. We see the highest variability of F_1 scores of cell line tasks in the absence of chemical, and in the F_1 scores of species in the absence of cell line. This is contrary to our expectation that similar entity types would perform consistently worse in each other's absence. It also may suggest that the effect of multitask training on task performance in this setting is not necessarily synergistic and is may be due to the robustness of the deep transformer model (BioBERT) and its ability to accommodate multiple tasks.

Table 6 shows the F_1 scores (evaluated on the test sets) resulting from the second finetuning, which was performed on the previously excluded tasks independently using the single task architecture. The *Ablation amount* column represents the proportion of sequences in the training set withheld during model training. For reference, STL and MTL learning results are shown in subsequent columns. Similarly to the multitasking models, hyperparameters used for this experiment are the same as with the single task models, and no additional hyperparameter tuning was performed other than the F_1 score-based model selection. For example, only 10% of the GPRO (gene) training dataset was needed to surpass its STL baseline. Its F_1 score at this ablation amount is equal to its F_1 score when multitask-trained with the full dataset collection (no ablation) and

[1] The scores for 0.0 ablation differ slightly from Tables 2 and 3 because they come from a separate runs with different random starting weights

Table 4
Single-task model data ablation results.

Entity	Dataset	Ablation amount 0.00	0.25	0.50	0.75	0.90
CELL LINE	CLL	0.748	0.793	0.729	0.727	0.726
	GELLUS	0.927	0.857	0.855	0.739	0.210
	JNLPBA	0.807	0.805	0.814	0.806	0.780
	average	0.828	0.818	0.799	0.757	0.572
CHEMICAL	CDR	0.937	0.934	0.924	0.919	0.908
	CEMP	0.914	0.911	0.913	0.908	0.899
	CHEBI	0.895	0.890	0.885	0.884	0.871
	CHEMDNER	0.947	0.945	0.941	0.934	0.919
	SCAI	0.941	0.937	0.929	0.913	0.899
	average	0.927	0.923	0.918	0.912	0.899
GENE	BC2GM	0.901	0.904	0.894	0.882	0.857
	BIOINFER	0.930	0.927	0.918	0.905	0.892
	DECA	0.731	0.715	0.716	0.710	0.631
	FSU	0.944	0.941	0.937	0.936	0.923
	GPRO	0.803	0.804	0.798	0.781	0.772
	IEPA	0.912	0.885	0.840	0.715	0.657
	JNLPBA	0.902	0.904	0.899	0.895	0.871
	MIRNA	0.760	0.742	0.709	0.599	0.328
	OSIRIS	0.800	0.779	0.766	0.748	0.749
	VARIOME	0.918	0.915	0.896	0.888	0.874
	average	0.860	0.852	0.837	0.806	0.755
SPECIES	LINNEAUS	0.633	0.697	0.598	0.620	0.088
	MIRNA	0.915	0.915	0.901	0.802	0.000
	S800	0.831	0.815	0.815	0.777	0.751
	VARIOME	0.789	0.667	0.000	0.535	0.724
	average	0.792	0.773	0.578	0.684	0.391
OVERALL AVERAGE		0.859	0.849	0.804	0.801	0.697

Table 5
Multitask model data ablation results.

Entity	Dataset	Ablation amount 0.00	0.25	0.50	0.75	0.90
CELL LINE	CLL	0.795	0.724	0.640	0.514	0.575
	GELLUS	0.824	0.711	0.744	0.222	0.862
	JNLPBA	0.743	0.787	0.806	0.801	0.765
	average	0.787	0.741	0.730	0.513	0.734
	STL average	0.828	0.818	0.799	0.757	0.572
CHEMICAL	CDR	0.947	0.940	0.937	0.936	0.924
	CEMP	0.915	0.914	0.912	0.918	0.911
	CHEBI	0.888	0.889	0.898	0.886	0.890
	CHEMDNER	0.941	0.942	0.939	0.941	0.914
	SCAI	0.826	0.937	0.917	0.921	0.917
	average	0.903	0.924	0.920	0.920	0.911
	STL average	0.927	0.923	0.918	0.912	0.899
GENE	BC2GM	0.875	0.867	0.887	0.886	0.857
	BIOINFER	0.920	0.909	0.916	0.907	0.899
	DECA	0.708	0.717	0.709	0.681	0.664
	FSU	0.926	0.934	0.933	0.927	0.905
	GPRO	0.800	0.805	0.798	0.802	0.773
	IEPA	0.809	0.755	0.737	0.745	0.697
	JNLPBA	0.881	0.892	0.891	0.887	0.876
	MIRNA	0.795	0.767	0.782	0.749	0.674
	OSIRIS	0.837	0.835	0.837	0.829	0.837
	VARIOME	0.918	0.922	0.921	0.904	0.869
	average	0.847	0.840	0.841	0.832	0.805
	STL average	0.860	0.852	0.837	0.806	0.755
SPECIES	LINNEAUS	0.721	0.450	0.471	0.825	0.000
	MIRNA	0.825	0.848	0.835	0.630	0.000
	S800	0.799	0.805	0.823	0.819	0.733
	VARIOME	0.689	0.606	0.583	0.713	0.000
	average	0.759	0.677	0.678	0.747	0.183
	STL average	0.792	0.773	0.578	0.684	0.391
OVERALL MTL AVERAGE		0.836	0.816	0.814	0.793	0.706
OVERALL STL AVERAGE		0.859	0.849	0.804	0.801	0.697

remains constant regardless of ablation amount ($sd = 0.003$). The same trend can be seen with CEMP and CHEMDNER. These three dataset are also the largest.

Fig. 6 shows the performance (F_1 score) of STL models that have been pre-trained using entity-ablated multitask learning, as well as MTL and the STL baseline models. The performances are grouped by entity type and the F_1 scores are averaged over models in that entity group. The bottom axis indicates the amount of training data removed before training. When considering these models at the entity class-level, the performance depends on the entity type. For the cell line class, it seems that STL is better suited, while species models benefit the most from pre-training on other biomedical entity types first (50% ablation and lower). One interesting observation is the S800 dataset. The dataset is abundant with species mentions in binomial nomenclature form (for example *E. coli*), which gives the most useful information when linking to a taxonomic datasource with a unique identifier. When 50% or more of the dataset is used for training, the model out-performs both the STL and MTL models.

7.5. Entity level comparison

In this section, we evaluate the STL and MTL models at the entity-level to allow for a direct comparison with previous works.

We first conduct a direct comparison with results reported by Weber et al. [10]. Weber et al. use an LSTM-CRF model for biomedical NER and compare gold standard pre-training (GSPT), silver standard pre-training (SSPT), and baseline models with no pre-training (No PT). For this evaluation, the IOB tags are used to determine the boundaries of an entity mention, and scores are computed using those boundaries. We evaluated our models such that a true positive requires that a mention must begin with a *B* label and subsequent labels with an *I*, and overlap exactly with the ground truth mention boundaries and labels.

Table 7 shows the entity-level evaluation of the MTL and STL baseline models. The table also includes a comparison to the results from Weber et al. shown to the right of the STL and MTL scores in each metric column. 11 of our STL models and 3 of our MTL models outperform the GSPT, SSPT, and No PT models in *F*1 scores. 10 of our STL models and 9 of our MTL models outperform all other models in precision. The GSPT, SSPT, and No PT models outperform our models in recall except 4 STL and 1 MTL.

Surprisingly, Linnaeus species has much lower *F*1 scores compared to the IOB tag/word-level evaluation. This suggests that the models are correctly identifying parts of the mentions, but not predicting the boundaries precisely. This may be mitigated by using a CRF on top of the encoder layers or using a method that does not rely on IOB labels to identify mention boundaries, as suggested by Sun et al. [12].

Lastly, we conduct a comparison with two previous proposed multitasking frameworks that evaluate their systems on four overlapping datasets, although, the tasks and data sources utilized in the frameworks vary. No previous work has evaluated multitasking for NER across biomedical articles at this scale. Table 8 shows the results of our STL and MTL models and the results reported by Peng, et al. [2] and Zuo, et al. [13]. The results show our MTL model obtained higher Precision, Recall and *F*1 scores except for BC2GM. In this case, Peng, et al. report higher scores than our MTL results but are on par with our STL results.

8. Related work

In this section, we describe previous works related to multitask learning and neural network-based approaches to BioNLP tasks.

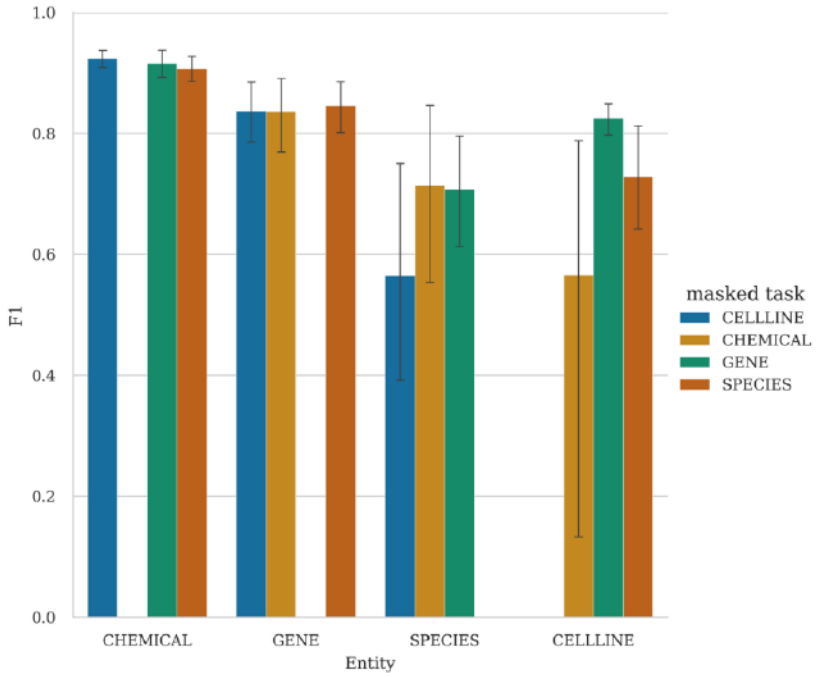


Fig. 5. Effect of entity ablation on MTL models. Four MTL models were trained in the absence of one entity type (entity ablation) indicated by bar-color. Bars show the average performance (F_1 score) of MTL models, grouped by entity. Error bars indicate within-entity standard error. Cell line models have relatively inconsistent performance when chemical entities are ablated compared to when gene entities are ablated.

Table 6
Learning new unseen entities using pretrained MTL models.

		Ablation amount					STL	MTL
Entity	**Dataset**	0.00	0.25	0.50	0.75	0.90	0.00	0.00
CELL LINE	**CLL**	0.605	0.548	0.546	0.519	0.500	0.885	0.795
	GELLUS	0.810	0.870	0.861	0.858	0.000	0.864	0.824
	JNLPBA	0.814	0.819	0.798	0.810	0.807	0.808	0.743
	average	0.743	0.746	0.735	0.729	0.436	0.852	0.787
CHEMICAL	**CDR**	0.936	0.935	0.927	0.930	0.912	0.934	0.947
	CEMP	0.912	0.912	0.913	0.912	0.909	0.913	0.915
	CHEBI	0.892	0.883	0.890	0.885	0.883	0.894	0.888
	CHEMDNER	0.947	0.945	0.945	0.945	0.934	0.944	0.941
	SCAI	0.947	0.941	0.944	0.931	0.918	0.939	0.826
	average	0.927	0.923	0.924	0.921	0.911	0.925	0.903
GENE	**BC2GM**	0.897	0.901	0.898	0.896	0.878	0.905	0.875
	BIOINFER	0.918	0.903	0.903	0.850	0.826	0.929	0.920
	DECA	0.720	0.719	0.720	0.722	0.704	0.711	0.708
	FSU	0.937	0.938	0.936	0.937	0.929	0.943	0.926
	GPRO	0.803	0.806	0.799	0.807	0.800	0.788	0.800
	IEPA	0.874	0.883	0.787	0.800	0.739	0.907	0.809
	JNLPBA	0.896	0.898	0.896	0.893	0.892	0.902	0.881
	MIRNA	0.734	0.670	0.718	0.625	0.031	0.747	0.795
	OSIRIS	0.774	0.778	0.797	0.612	0.361	0.812	0.837
	VARIOME	0.926	0.920	0.925	0.914	0.776	0.924	0.918
	average	0.848	0.842	0.838	0.806	0.694	0.857	0.847
SPECIES	**LINNEAUS**	0.865	0.870	0.854	0.794	0.000	0.771	0.721
	MIRNA	0.878	0.868	0.877	0.000	0.000	0.906	0.825
	S800	0.828	0.823	0.825	0.809	0.778	0.831	0.799
	VARIOME	0.768	0.759	0.681	0.615	0.000	0.548	0.689
	average	0.835	0.830	0.809	0.554	0.195	0.764	0.759
OVERALL AVERAGE		0.849	0.845	0.838	0.776	0.617	0.855	0.836

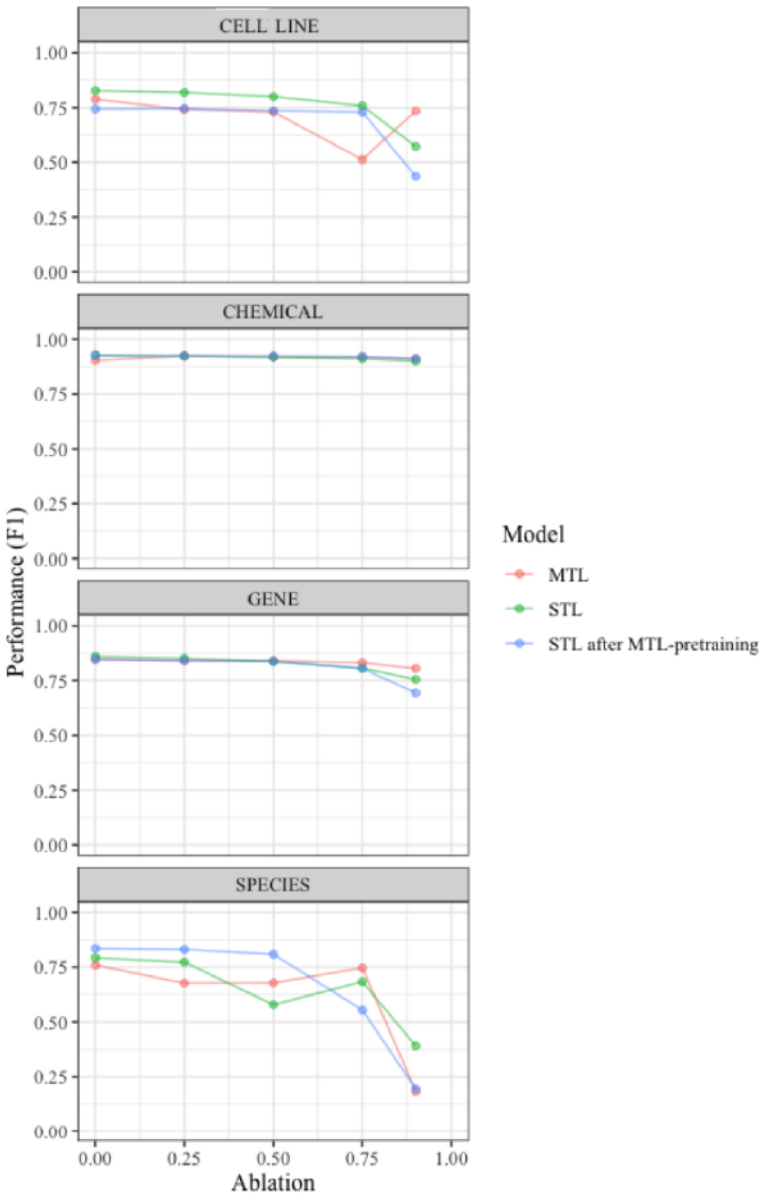


Fig. 6. Effect of MTL pre-training on STL model performance. Performance of STL learning models pretrained using MTL with entity ablation, grouped by entity. MTL models were first trained in the absence of datasets of one entity type, then finetuned on a single task of the previously ablated entity type.

Crichton et al. [14] demonstrated that multitask learning can improve the performance on NER tasks compared to single task learning in the biomedical domain. They used static word embeddings in a lookup table fed into a convolution layer, a fully connected layer and finally into a softmax output layer for the final classification. We adopt a similar architecture as their 'multi-output multitask convolution model'. However, our models replace the static word embeddings and convolution layer with a transformer-based embedding layer [5] which produces contextualized subword embeddings. For simplicity, we choose to adopt the same output layer, as opposed to the CRF output layer used by Akdemir and Shibuya [3] who proposed a single task model using a BioBERT-CRF architecture. Other architectures, including the BiLSTM-CRF have also been used for NER [15].

Akdemir and Shibuya [3], and Peng and Chen [2] proposed a pairwise multitask learning design to investigate the effect that tasks have on each other's performance when jointly trained. Our approach instead considers each entity type as a task group, and permutes the entity ablation across each task individually. Furthermore, we simulate the real-world scenario of adding new unseen entity types/tasks to the multitask model, and observe the performance of the new tasks after removing various proportions of training data.

Multitask models often share representations between their task-specific layers, as in Crichton et al.[14]. However, we chose not to share task-specific representations, but instead hypothesized that multitask transformer encoder training would be sufficient to observe comparable performance compared to single task-trained models. Although the models evaluated in this work are all performing NER, we consider them to be separate tasks because the meaning of the labels can vary between datasets within the same entity type group, as in the case of cell line entities whose mentions can include long noun phrases as opposed to a gene whose mentions often span one or two nouns. Similarly, species tasks vary in the usage of biological nomenclature in their annotations. For example species mentions in the MIRNA dataset are often common organism names, as opposed to the more granular binomial nomenclature which may be more useful to researchers in a biological domain like microbiology.

Methods have been developed with the similar goal of reducing the required resources for annotation in text mining, for example, active

Table 7
Comparison to Weber et al. (2020) [10].

		F1					Precision					Recall				
Entity	**Dataset**	**MTL**	**STL**	**No PT**	**GSPT**	**SSPT**	**MTL**	**STL**	**No PT**	**GSPT**	**SSPT**	**MTL**	**STL**	**No PT**	**GSPT**	**SSPT**
CELL LINE	**CLL**	0.708	**0.816**	0.790	0.730	0.800	0.896	0.922	0.753	0.761	0.822	0.585	0.732	0.831	0.701	0.779
	GELLUS	0.743	**0.822**	0.682	0.714	0.805	0.802	0.794	0.873	0.829	0.888	0.692	**0.852**	0.559	0.628	0.737
	JNLPBA	0.618	0.669	0.673	0.649	0.671	0.661	0.692	0.695	0.657	0.730	0.581	0.648	0.654	0.641	0.620
CHEMICAL	**CDR**	**0.933**	**0.932**	0.907	0.929	0.921	0.947	0.945	0.919	0.935	0.918	0.919	0.920	0.896	0.923	0.925
	CEMP	0.852	0.848	0.856	0.855	0.863	**0.893**	**0.892**	0.823	0.832	0.842	0.815	0.808	0.891	0.879	0.886
	CHEBI	0.733	0.760	0.764	0.804	0.786	0.784	0.773	0.754	0.833	0.798	0.688	0.747	0.775	0.776	0.775
	CHEMDNER	**0.907**	**0.915**	0.884	0.889	0.883	**0.908**	**0.919**	0.893	0.905	0.887	**0.907**	**0.911**	0.877	0.873	0.878
	SCAI	0.744	0.750	0.689	0.778	0.774	**0.811**	**0.832**	0.726	0.750	0.782	0.687	0.682	0.656	0.808	0.765
GENE	**BC2GM**	0.776	**0.819**	0.780	0.779	0.786	0.792	**0.821**	0.780	0.800	0.796	0.760	**0.818**	0.780	0.759	0.776
	BIOINFER	0.831	0.855	0.840	0.846	0.859	0.814	**0.867**	0.858	0.860	0.851	0.849	0.843	0.822	0.833	0.868
	DECA	**0.731**	**0.729**	0.689	0.688	0.690	**0.769**	**0.784**	0.656	0.661	0.646	0.696	0.681	0.725	0.719	0.740
	FSU	0.862	**0.888**	0.881	0.884	0.879	**0.887**	**0.916**	0.874	0.880	0.874	0.838	0.862	0.887	0.888	0.885
	GPRO	0.714	**0.730**	0.703	0.718	0.720	**0.799**	**0.844**	0.646	0.657	0.664	0.646	0.643	0.771	0.791	0.786
	IEPA	0.832	0.869	0.861	0.824	0.894	0.857	0.870	0.879	0.808	0.875	0.808	0.867	0.843	0.840	0.913
	JNLPBA	0.805	**0.827**	0.818	0.805	0.823	**0.846**	**0.835**	0.806	0.791	0.802	0.768	0.819	0.830	0.819	0.845
	MIRNA	0.669	0.683	0.634	0.697	0.707	0.622	**0.797**	0.712	0.644	0.678	0.724	0.598	0.570	0.760	0.739
	OSIRIS	0.860	0.747	0.816	0.874	0.860	0.873	0.760	0.840	0.846	0.847	0.847	0.735	0.794	0.904	0.873
	VARIOME	0.930	**0.944**	0.942	0.941	0.939	**0.945**	**0.959**	0.935	0.930	0.924	0.916	0.930	0.950	0.952	0.955
SPECIES	**LINNEAUS**	0.269	0.651	0.856	0.932	0.936	0.680	0.543	0.921	0.945	0.949	0.168	0.812	0.799	0.920	0.923
	MIRNA	0.791	0.897	0.895	0.909	0.864	0.811	0.841	0.933	0.919	0.908	0.773	**0.960**	0.859	0.899	0.824
	S800	0.703	**0.737**	0.711	0.725	0.716	0.729	**0.776**	0.722	0.752	0.730	0.679	0.702	0.701	0.701	0.703
	VARIOME	0.556	0.692	0.732	0.701	0.762	**0.985**	**0.955**	0.644	0.614	0.691	0.387	0.543	0.849	0.818	0.849

Table 8
Comparison to Peng, et al. [2] and Zuo, et al. [13].

	F1				*Precision*				*Recall*			
Dataset	*MTL*	*STL*	[13]	[2]	*MTL*	*STL*	[13]	[2]	*MTL*	*STL*	[13]	[2]
CDR	0.933	0.932	0.889	0.931	0.947	0.945	0.894	–	0.919	0.920	0.883	–
CHEMDNER	0.907	0.915	0.886	0.729	0.908	0.919	0.907	–	0.907	0.911	0.886	–
BC2GM	0.776	0.819	0.821	-	0.792	0.821	0.819	–	0.760	0.818	0.822	–
JNLPBA	0.805	0.827	0.742	–	0.846	0.835	0.708	–	0.768	0.819	0.779	–

learning [16]. Although they may compliment the methods proposed here, they are outside the scope of this work.

9. Conclusions and future work

In this work, we found that in practice, single task NER modeling works well when the number of tasks is relatively small. However, as the number of tasks in an NLP pipeline increase, so does the combined size of all models. This can be problematic especially when using hardware with limited resources. Therefore, it may be beneficial to reduce the number of models used in an NER pipeline with minimal trade-off with task performance.

Our data ablation experiments demonstrate that multitasking models for biological NER can perform well with only a fraction of the training data in available gold-standard datasets, but in most cases with some decrease in performance compared to single-task models trained with the same hyperparameters. A trade-off to this approach is training time. Depending on the sampling method during training, an unbalanced dataset collection can result in much longer training time compared to STL.

Our entity ablation experiments that MTL can be updated with an unseen entity without a significant reduction in performance compared to MTL and STL.

Future work includes examining how well MTL models can be updated over time. For example, if the initial models are trained using full gold-standard datasets, and over time they are updated with small amounts of training data for new task learning, is it necessary to retrain the models from scratch including all previous training data, or simply update the model on a single task at a time?

We would also like to examine the efficacy of multitask models for inference on unlabelled datasets like research articles. In practice, we observe single task models to be sensitive to the document types of the training set and document on which inference is performed. For example, models trained only on abstracts produce erroneous predictions that go unnoticed during the model evaluation. Although evaluating on a different test set or combining training sets could potentially mitigate this problem during model selection, multitasking amy offer an alternative method for which we can improve inference performance.

Mulyar and McInnes [7] chose to share information in the task-specific layers. This approach more directly addresses an objective addressed here of leveraging information within hidden layers to boost performance. However, since the current work also addresses the question of how much data is needed to produce useful representations for downstream tasks, we chose to isolate the effect of the encoder layers, which in theory have a greater capacity to represent information than the task-specific layers given the larger number of trainable parameters. Furthermore, we will investigate how much different classification layers and tasks perform, such as CRFs used by Akdemir and Shibuya [3], as well as relation extraction tasks in the biological and clinical domains.

Finally, we would like to investigate the potential benefits of optimizing of the multitasking models, including warm-up steps on datasets we observe to have either strongly fluctuating performance during training or vary the learning rate on datasets that tend to train more slowly than others.

Funding

This work was funded by the National Science Foundation under Grant No. 1939951.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank Gaurav Nakum and Andriy Mulyar for their help gathering and processing the 22 datasets used here. We'd also like to thanks Andriy for his support while adapting the mutlitasking framework for use in these experiments.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.jbi.2022.104062.

References

[1] D. Nadeau, S. Sekine, A survey of named entity recognition and classification, Lingvisticae Investigationes 30 (1) (2007) 3–26.
[2] Y. Peng, Q. Chen, Z. Lu, An Empirical Study of Multi-Task Learning on BERT for Biomedical Text Mining, arXiv:2005.02799 [cs]. URL: http://arxiv.org/abs/2005.02799.
[3] A. Akdemir, T. Shibuya, Analyzing the Effect of Multi-task Learning for Biomedical Named Entity Recognition, arXiv:2011.00425 [cs]ArXiv: 2011.00425. URL: http://arxiv.org/abs/2011.00425.
[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is All you Need, Adv. Neural Inform. Process. Syst. 30 (2017) 5998–6008.
[5] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv: 1810.04805.
[6] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C.H. So, J. Kang, Biobert: a pre-trained biomedical language representation model for biomedical text mining, Bioinformatics. https://doi.org/10.1093/bioinformatics/btz682.
[7] A. Mulyar, O. Uzuner, B. McInnes, MT-clinical BERT: scaling clinical information extraction with multitask learning, J. Am. Med. Informat. Assoc. 28 (10) (2021) 2108–2115, https://doi.org/10.1093/jamia/ocab126, https://academic.oup.com/jamia/article-pdf/28/10/2108/40408880/ocab126.pdf.

[8] H. Cho, H. Lee, Biomedical named entity recognition using deep neural networks with contextual information, BMC Bioinformat. 20 (1) (2019) 735, https://doi.org/10.1186/s12859-019-3321-4. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3321-4.

[9] K. Hakala, S. Pyysalo, Biomedical Named Entity Recognition with Multilingual BERT, in: Proceedings of The 5th Workshop on BioNLP Open Shared Tasks, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 56–61. https://doi.org/10.18653/v1/D19-5709. URL: https://aclanthology.org/D19-5709.

[10] L. Weber, J. Münchmeyer, T. Rocktäschel, M. Habibi, U. Leser, Huner: improving biomedical ner with pretraining, Bioinformatics 36 (1) (2020) 295–302.

[11] J. Baldridge, The opennlp project, URL: http://opennlp.apache.org/index.html (accessed 2 February 2012) (2005) 1.

[12] C. Sun, Z. Yang, L. Wang, Y. Zhang, H. Lin, J. Wang, Biomedical named entity recognition using bert in the machine reading comprehension framework, J. Biomed. Inform. 118 (2021) 103799.

[13] M. Zuo, Y. Zhang, Dataset-aware multi-task learning approaches for biomedical named entity recognition, Bioinformatics 36 (15) (2020) 4331–4338.

[14] G. Crichton, S. Pyysalo, B. Chiu, A. Korhonen, A neural network multi-task learning approach to biomedical named entity recognition, BMC Bioinformat. 18 (2017) 368, https://doi.org/10.1186/s12859-017-1776-8.

[15] X. Wang, Y. Zhang, X. Ren, Y. Zhang, M. Zitnik, J. Shang, C. Langlotz, J. Han, Cross-type biomedical named entity recognition with deep multi-task learning, Bioinformatics 35 (2019) 1745–1752, https://doi.org/10.1093/bioinformatics/bty869.

[16] A. Agrawal, S. Tripathi, M. Vardhan, Active learning approach using a modified least confidence sampling strategy for named entity recognition, Prog. Artif. Intell. https://doi.org/10.1007/s13748-021-00230-w.