

Project Report

Named Entity Recognition of Different Granularities

Formal Semantics, Winter Semester 24/25

Julian Partanen (4204806), Paul Kupper (4736288), Thomas Wolf
(4273318)

Heidelberg, 31. March 2025

Abstract

Named Entity Recognition (NER) and Named Entity Classification (NEC) aren't new problems and have many existing and performant solutions. In this project however we try to solve NEC with pre-trained generic transformer models like T5 and different LLMs by translating the problem into a task that is more familiar to these models like natural language inference (NLI), masked language modeling (MLM) or prompting. We compare our different task formulations with each other in both a zero-shot setting as well as after fine-tuning the models and we also compare their performance to a state-of-the-art solution (GLiNER). We also try to gain as much introspection as possible into these models to understand which parts of the input and task formulation play the most important role in these settings.

Outline

Abstract	2
1 Introduction	1
1.1 Named Entity Recognition	1
1.2 Named Entity Classification	1
1.3 State of the Art	1
2 Models and Datasets	2
2.1 Models	2
2.1.1 T5	2
2.1.2 GLiNER	2
2.1.3 Large Language Models (LLMs)	2
2.1.3.1 Llama-3.1-8B	2
2.1.3.2 DeepSeek-R1-Distill-Qwen-32B	3
2.2 Datasets	3
2.2.1 CoNLL	3
2.2.2 Pile-NER	3
2.2.3 FIGER	4
3 Method	6
3.1 GLiNER NEC	6
3.2 NLI with T5	6
3.3 MLM with T5 (label masking)	7
3.4 MLM with T5 (entity masking)	8
3.5 Fine-tuning	8
3.6 Word2Vec	9
3.7 LLM Prompting	9
4 Experiments	11
4.1 Model Comparison	11
4.2 Analysing Context Word Relevance	11
4.2.1 Contextless Classification	11
4.3 Hypotheses	11
5 Results and Analysis	12
5.1 GLiNER NER	12
5.2 NEC Overview	13
5.2.1 T5 zero-shot	13
5.2.2 T5 fine-tuned	14
5.2.3 LLM prompting	16
5.2.3.1 NER	16
5.2.3.2 NEC	16
5.3 Dataset Reliability	17
5.3.1 CoNLL	17
5.3.2 FIGER-coarse	17
5.3.3 FIGER-fine	18

5.3.4	Compensating for the Dataset Problems	18
5.4	Context Relevance	19
5.4.1	Relevance of specific words in the input sentence	19
5.4.2	Relevance of the surrounding context compared to the entity itself ..	20
6	Discussion and Outlook	22
6.1	NLP T5 task formulations	22
6.2	Regarding our hypotheses	22
6.3	Competitiveness of Large Language Models in NER tasks	23
6.4	Poor dataset quality	23
	Bibliography	24
	Individual Author Contributions	26
A	Appendix	27
A.1	FIGER - list of all labels	27
A.2	Accuracy after 100 Epochs of Fine-tuning	27
A.3	Examples from the Dataset Reliability Section	27
A.3.1	GLiNER	28
A.3.2	FIGER-coarse	28
A.3.3	FIGER-fine	29

1 Introduction

In this project, we aim to compare various approaches to named entity recognition and named entity classification. We evaluate based on the achieved accuracies and analyze the effect of model fine-tuning. We also perform an analysis of the relevance of the context for entity classification.

1.1 Named Entity Recognition

Named entity recognition (NER) is a problem in computer linguistics in which entities must be identified within a given text or sentence and then classified into one of several possible labels. These labels may for example be *person*, *building*, or *location*. Besides serving as a benchmark for natural language processing techniques, NER also has practical applications when extracting information from unstructured text. This may for example be useful in search engines, chat-bots, and other real-world tasks. Today, NER techniques generally rely on deep learning to achieve competitive results.

1.2 Named Entity Classification

Named entity classification (NEC) is a simplified formulation of the NER problem. Whereas in NER, the position of the entities in the sentence is unknown, named entity classification only requires the classification of a given entity in the input text whose span is already known. As we will see in Section 3, this enables several interesting approaches that would be impractical with an NER task formulation. The NEC task formulation also avoids potential ambiguities in the spans of entities. For example, it may be unclear whether quantifiers, adjectives, etc. are part of an entity, which makes evaluating the correctness of a prediction more difficult. In this work, we therefore mainly focus on the NEC task setting.

1.3 State of the Art

Transformer-based techniques such as GLiNER (Zaratiana et al. 2023) have achieved high accuracies in classical named entity recognition, with the achieved F1 scores in task settings, such as CoNLL, in some cases surpassing human-level performance (Tjong Kim Sang and De Meulder 2003). In our evaluation, we therefore chose GLiNER adapted to the NEC task as our reference benchmark in order to compare our own techniques. We discuss GLiNER in more detail in Section 2.1.2.

2 Models and Datasets

In the following sections we introduce the models and datasets we used in this project.

2.1 Models

2.1.1 T5

T5 is an encoder/decoder text-to-text transformer model introduced by Google (Raffel et al. 2023). It closely follows the originally proposed form of the transformer model described in the paper ‘Attention Is All You Need’ (Vaswani et al. 2023). The goal of T5 is to teach the model generalized knowledge about taking text as input and producing new text as output, and to then use that knowledge for specialized downstream tasks. This is also called transfer learning and gives T5 its name (**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer). The generalized knowledge was taught in pre-learning using techniques like corrupting spans (similar to masked language modeling that was used to train BERT, more on this in Section 3.3). Some examples of the downstream Natural Language Processing (NLP) tasks are translation, evaluating grammaticality of sentences (COLA), evaluating semantic similarity of sentences (STSB), summarizing text, natural language inference (NLI, see Section 3.2) and more. All of these downstream tasks are formulated as pure text inputs to the model.

2.1.2 GLiNER

In this work, we will use GLiNER (Zaratiana et al. 2023) as our reference model for comparison with our own approaches. GLiNER, which stands for ‘Generalist Lightweight Model for Named Entity Recognition’, is an advanced model designed to identify various types of entities within text using a bidirectional transformer encoder, akin to models like BERT (Devlin et al. 2018). It offers a practical alternative to traditional NER models that are confined to predefined entity categories and to large language models (LLMs) that, despite their flexibility, are often resource-intensive and costly. A huge advantage making this model particularly useful for us is that the labels are not predefined, enabling GLiNER to work with arbitrary sets of labels (of limited size) without having to ‘learn’ those labels first. GLiNER was trained on the Pile-NER Dataset (Section 2.2.2).

2.1.3 Large Language Models (LLMs)

In this work, we additionally evaluate the suitability of LLMs for NER tasks. We chose to focus on Llama-3.1-8B (Grattafiori et al. 2024) and DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI 2025) since the Llama model is the most obvious representative of a lightweight model we can run locally and the DeepSeek model of the modern reasoning models and both are open-source and accessible for free.

2.1.3.1 Llama-3.1-8B

Llama-3.1-8B (Grattafiori et al. 2024) is a multilingual LLM developed by Meta AI that features 8 billion parameters. Released in July 2024 as part of the Llama 3.1 series,

it is designed for lightweight, efficient deployment across various platforms. The 8B variant is particularly noted for its balance between performance and resource efficiency, making it suitable for local text generation tasks and enabling us to run it locally via Ollama.

2.1.3.2 DeepSeek-R1-Distill-Qwen-32B

DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI 2025) is a distilled reasoning LLM developed by DeepSeek-AI, based on the Qwen 2.5 architecture with 32 billion parameters. What sets DeepSeek apart is its innovative use of reinforcement learning (RL) to enhance reasoning capabilities without relying on supervised fine-tuning. This approach allows the model to develop complex problem-solving skills, such as self-verification and reflection, through pure RL methods.

2.2 Datasets

	CoNLL	Pile-NER	FIGER
Size (sentences)	14041	45889	2004220
Number of distinct Labels	4	12963	<ul style="list-style-type: none"> • coarse: 7 • fine: 106
Annotation Reliability	low (Rücker and Akbik 2023)	medium ¹	very low ¹

Table 1: Dataset Overview

2.2.1 CoNLL

The CoNLL-03 dataset, introduced during the 2003 Conference on Natural Language Learning (CoNLL) (Tjong Kim Sang and De Meulder 2003) comprises annotated text from the Reuters Corpus, focusing on four entity types: persons (PER), organizations (ORG), locations (LOC), and miscellaneous entities (MISC). Recent studies have identified annotation inconsistencies and errors within the dataset (Rücker and Akbik 2023).

2.2.2 Pile-NER

The Pile-NER dataset, introduced in 2023 by Zhou et al. (Zhou et al. 2023), is a comprehensive resource for open-type Named Entity Recognition (NER). Derived from the Pile Corpus, it encompasses approximately 240,000 entity mentions and 12963 distinct entity types (after lowercasing all the labels). The dataset’s passages are enhanced through processing with ChatGPT, facilitating the transparent generation of entity annotations. Due to the diversity of labels, this dataset is most suitable for training models such as GLiNER (Section 2.1.2). The labels are mostly English, but upon closer inspection consist of all sorts of languages (Arabic, Chinese, German, Russian, Spanish, ...). Many labels are highly specific (e.g. ‘2,4,6-trimethylphenyl’) or strongly overlap (e.g. ‘mythical being’, ‘mythical character’, ‘mythical creature’, ‘mythical entity’, ‘mythical

¹See Section 5.3

figure'). These peculiarities make the Pile-NER dataset less suitable for the evaluation of our approaches, not to mention the fact that some of the methods discussed in this work are inefficient for large numbers of labels. The latter problem could perhaps be mitigated by providing a fixed number of possible labels containing the true label for each test instance.

2.2.3 FIGER

Fine-Grained Entity Recognition (FIGER) is a NER system with higher granularity than many alternatives that were only trained on datasets like CoNLL with only 4 labels (Ling and Weld 2021). This system was trained on a custom dataset with 112 tags in total generated from the Freebase types set. Even though the FIGER dataset has been curated by hand, this still poses some possibility for errors in the dataset since these labels are community-assigned. In this work, when we mention FIGER, we refer to the dataset, not to the NER system.

The main upside of the FIGER dataset is that it is a lot more fine-grained than CoNLL, but without having thousands of labels like Pile-NER thus bridging the gap between these two datasets. Our approaches (NLI and MLM task formulations, Section 3) don't scale well with an increased amount of labels since for example for NLI we formulate one hypothesis for each label in the dataset to classify an entity. Because of this, FIGER is our best option to test a more fine-grained NEC setting while still being feasible for our custom approaches.

Another opportunity that FIGER provides is that it is a hierarchic dataset, meaning that each labeled entity has a coarse and a fine label. After some modifications and cleanups to the dataset, we were able to provide two sets of labels for FIGER:

- A coarse one containing 7 labels: person, organization, location, product, art, event and building
- A fine one containing 106 labels: actor, architect, artist, athlete, ... (Section A.1)

This allows us to isolate testing of label granularity without any other variables, for example, we can provide our model with the same subset of the FIGER dataset with the same sentences and entities to categorize, but one time with the coarse label set and a second time with the fine label set. Any differences in performance between these two runs should only reflect on the models' capability to handle fine-grained labels. Here is an example of how the labels might change with granularity:

FIGER-coarse:

Sentence: In 1995, Martin appeared on Rosie Flores 's Rockabilly Filly album for HighTone Records.

Entity: Rosie Flores

True Label(s): person

FIGER-fine:

Sentence: In 1995, Martin appeared on Rosie Flores 's Rockabilly Filly album for HighTone Records.

Entity: Rosie Flores

True Label(s): musician, artist

FIGER, especially FIGER-fine can have multiple labels per entity. A model only needs to predict one of the true labels for it to be evaluated as correct. Only one entity per sentence is labeled.

3 Method

We evaluate several methods for named entity classification, focusing on Google’s T5 model but also investigating alternatives such as Large Language Models. We evaluate how relevant the sentence context is for the model to make its decision compared to performing the classification of the entity by itself.

3.1 GLiNER NEC

GLiNER does not natively support named entity classification, meaning we cannot specify a target entity within a sentence to classify. To apply GLiNER to the NEC task anyway, it can be adapted by extracting entities from the text (NER) and matching them to the target entity. The following pseudocode details this approach:

```
def NEC(sentence, target_entity, labels):
    labeled_entities = NER(sentence, labels)
    for entity in labeled_entities:
        if entity['text'] == target_entity:
            return entity['label']
    return "Target entity not found during NER."
```

Obviously, this approach is at a disadvantage compared to native NEC approaches that can take advantage of the target entity being provided and will take significant losses in accuracy due to not finding the correct entity in the first place. Nevertheless, this will serve us as a comparison point that useful NEC models should be competitive against.

3.2 NLI with T5

As our first approach, we formulate the named entity classification task as a set of natural language inference tasks. Given a sentence, the span of the entity within the sentence, that is to be classified, and the set of possible labels we create a set of hypotheses, one for each label.

To illustrate the task setting, we provide an example. Given the sentence ‘*Barack Obama was the president of the United States.*’, with ‘*Barack Obama*’ being the entity to be labeled and the possible labels being *organization*, *location*, and *person*, we create the following hypotheses:

- Barack Obama was the president of the United States. Barack Obama is a organization
- Barack Obama was the president of the United States. Barack Obama is a location
- Barack Obama was the president of the United States. Barack Obama is a person

These hypotheses are then given to the NLI model. Initially, we intended to use `google/t5_xxl_true_nli_mixture`, a variant of Google’s T5 model, fine-tuned for natural language inference tasks as described in (Honovich et al. 2022). Due to memory constraints of the GPUs available to us, we resorted to using the smaller `google/t5_base` variant, which is also trained on a set of multiple downstream tasks including NLI, and still

performs reasonably well in this task. We expect to recover some of the accuracy lost by using the base model that is not exclusively optimized for NLI through our own fine-tuning as described in Section 3.5.

We then look at the probability of entailment for each of the hypotheses and pick the hypothesis with the highest probability of entailment for our prediction. The label used in this hypothesis is the label predicted for the entity to be classified.

One shortcoming of this approach is the grammaticality of the hypothesis. As the label is inserted verbatim into the hypothesis, ungrammatical hypotheses may occur. In our example ‘*Barack Obama is a organization*’ should ideally be ‘*Barack Obama is **an** organization*’. There are several possible approaches to mitigate this issue. Initially, we will evaluate the output quality without any mitigations. In a later step, we make use of fine-tuning, to optimize the model to our task formulation. We hypothesize that this process is able to mitigate this possible issue as the model may become accustomed to receiving ungrammatical input.

3.3 MLM with T5 (label masking)

Our second approach formulates the NEC task as a masked language modeling (MLM) task. For each sentence and entity in the input sentence, we will formulate a hypothesis similar to the hypotheses in our NLI approach. Instead of a hypothesis for each label, however, we just have one hypothesis with a mask at the position where the label should be. We then present the model with a list of all labels and return the one with the highest probability of replacing the mask.

- Barack Obama was the president of the United States. Barack Obama is a [MASK].
 - Try the labels ‘organization’, ‘location’, and ‘person’ and return the one that best fits into the gap

Contrary to BERT, T5 wasn’t trained on MLM but on a slight variation called corrupting spans. The main difference for our purposes is that T5 allows spans to replace multiple tokens of text while Bert’s MLM only masks one token at a time. This means that because we use T5 this approach also allows for multi-token labels. The task formulation for T5’s span corruption looks a bit more complex but is essentially the same:

- Barack Obama was the president of the United States. Barack Obama is a <extra_id_0>.
 - Compute loss for outputs ‘<extra_id_0> organization <extra_id_1>’, ‘<extra_id_0> location <extra_id_1>’, ... and return the one with the lowest loss

As with the NLI task formulation, we make use of the `google/t5_base` model to limit memory usage. This approach has the same shortcomings as our NLI approach since

it also suffers from the possibility of ungrammatical hypotheses. We will rely on fine-tuning to mitigate this for this approach as well.

3.4 MLM with T5 (entity masking)

Our third and final T5-based approach is also an MLM task formulation. This time however we don't add a hypothesis with a masked label to the sentence but instead mask the entity itself in the original sentence. We then query the probabilities of different label representatives from the same dataset (chosen at random, current entity excluded) and choose the label that the representative with the highest probability belongs to.

- [MASK] was the president of the United States.
 - Try the representatives United Nations, Nestlé, Tokyo, Hudson River, Keanu Reeves, Angela Merkel, ... and return the label that the representative with the highest probability belongs to

For this approach, we highly rely on T5's ability to mask multiple tokens since most entities consist of more than one token. As before, we are using the `google/t5_base` model.

We expect this approach to perform significantly worse than our first two approaches since by masking the original entity the model gets less information as input. Previously the model had both the entity and the context around it as input, now it only has the latter. However, this experiment might still prove useful for learning about the importance of the sentence context compared to the entity itself.

Another disadvantage is that the success of this approach is highly dependent on the choice of representatives for each class. In the example above, if one of the randomly chosen representatives for the person class had been 'Donald Trump' instead the model would probably assign it a much higher probability compared to 'Keanu Reeves' or 'Angela Merkel' since both have never been the president of the United States. This might only improve to a certain degree by fine-tuning the model since during training the model only gets presented by one representative as the correct replacement for the mask even though all representatives of the gold label would have been correct.

3.5 Fine-tuning

Up to this point we have used existing models trained on generic text corpora with more general task formulations such as NLI and MLM. In an attempt to improve the accuracy of our models, we now employ fine-tuning. From our datasets, we create a set of task formulations on which we further train each model. We hypothesize that this can greatly improve the accuracy of each model as compared to zero-shot approaches with the generic models.

For both the T5 NLI and the T5 MLM with class masking, we extract 1000 sentences from the FIGER-coarse dataset and construct the tasks for each pre-labeled entity. Depending on the method, this results in between 1221 and 8547 tasks. We then train

the model, starting from the pre-trained baseline `google/t5_base`, for 100 epochs. Model checkpoints are saved after each epoch.

3.6 Word2Vec

For the task of named entity classification, it may be interesting to know how much of the information used in the classification comes from the entity itself, as opposed to the context it is placed in. We therefore investigate a classification approach using word embeddings. This approach allows us to eliminate any context information and therefore to investigate the degree of classification accuracy that can be achieved from the entity alone. Word2Vec is a word embedding method for turning words into n -dimensional vectors that statistically capture some semantic relations between words. They are obtained by training a neural network model on a large text corpus to either predict the target word given its surrounding context or alternatively predict the context given the target word (Mikolov et al. 2013). The trained weights are used to obtain embeddings for each word. One of the semantic properties that are captured is the semantic similarity which can be obtained by computing the cosine distance between two word embeddings (Mikolov et al. 2013). A high similarity shows that the words frequently occur in similar contexts and are therefore likely semantically similar.

When restricting the named entity recognition task to only single-word entities for which the embeddings are known, we can construct a basic named entity classification formulation. For each class label, we pick a set of representative entities that fit the given label. For example, for the label *location* we may pick the representatives *Canada*, *forest*, and *8th street*. The number of representatives must be high to cover a wide range of different entities matching the label.

In order to perform the classification, we then determine the word embedding for the entity to be classified and compute the similarity of this embedding with the embeddings of all of the representative words. The closest matching representative is determined and its label is used as the prediction.

This method has the obvious downside of only being able to handle known words for which embeddings have already been determined.

We evaluate both using a pre-trained Word2Vec model trained on the Google News corpus (Mikolov et al. 2013), as well as training our own model on our chosen NER datasets. As our initial tests with the self-trained Word2Vec model show very little promise, likely due to the small amount of training data available, this report only includes the results from the pre-trained Word2Vec model. The implementation for the self-trained Word2Vec model is nevertheless included in the projects code repository.

3.7 LLM Prompting

Due to their outstanding generalization capabilities, large language models have gained widespread use for a wide variety of tasks in recent years. We therefore compare our more targeted approaches shown in previous sections with a simple prompting approach

of several large language models. A prompt containing the task description and an example are passed to the LLM. Since LLMs tend to output unnecessary tokens (e.g. explanations), we need to extract the answer. To make the answer easier to extract, the prompt contains instructions on how to mark the answer. The prompt used in this project for named entity classification looks like this:

```
"You are part of a named entity classification pipeline.  
Given an entity, find the best fitting label of the provided labels (do not  
invent your own labels!).  
Choose the label so that the sentence "Target entity is a chosen_label" makes  
the most sense.  
Mark your result like this for easy extraction: <answer>predicted_class</answer>.
```

Example:

```
Labels == ['person', 'organization', 'location', 'miscellaneous']  
Sentence: 'NASA sent astronauts to the moon.'  
Target Entity: NASA  
Desired Result: <answer>organization</answer>
```

Your Task:"

Then the actual available labels, the target sentence, and the target entity are added to the prompt.

We also provide a similar implementation for named entity recognition, where the model first has to find the entities to classify.

A disadvantage of this approach is that the accuracy is expected to go down significantly for a larger number of possible labels. Furthermore, large language models, especially reasoning models, can be slow and resource-intensive. Nevertheless, we hypothesize that this approach will deliver results of high accuracy, making it competitive with state-of-the-art NEC methods.

4 Experiments

4.1 Model Comparison

We evaluate the model variants shown in Section 3 on their NEC accuracy on 100 test instances from the CoNLL, FIGER-fine, and FIGER-coarse datasets. For the fine-tuned model variants we ensure that there is no overlap in the examples from the dataset that are used for fine-tuning and those which are used for model evaluation. The results can be found in Table 3.

We additionally analyze how the fine-tuning of a model on one of our datasets transfers to other datasets.

GLiNER is also evaluated on the NER task, since the NEC implementation with GLiNER that serves as our benchmark is based on NER as detailed in Section 3.1.

4.2 Analysing Context Word Relevance

As we are interested in the importance of context for the named entity classification task, we perform an analysis where we replace individual words in the context with a placeholder *[blank]* and determine the effect of the prediction. We do this for each individual word in the input sentence including those which are part of the entity in which is to be classified.

4.2.1 Contextless Classification

As an extreme case, we use the Word2Vec embeddings method as described in Section 3.6 to obtain a baseline on the contextless classification of single-word entities.

4.3 Hypotheses

We had three main hypotheses regarding our NLP task formulations for T5 that we tried to prove or disprove with our experiments. First, we stated that fine-tuning would be required to achieve good classification performance, especially for more fine-grained datasets like FIGER-fine.

We also hypothesized that our Word2Vec baseline would benefit from finer granularity since the word embeddings would then be closer, whereas our T5 approaches would rather benefit from coarser labeled datasets.

Our final hypothesis was that the performance improvements gained by fine-tuning the T5 models wouldn't translate to datasets with different label names, especially if those datasets use more fine-grained labels.

5 Results and Analysis

In the following sections, we evaluate and interpret the results of our experiments.

5.1 GLiNER NER

	CoNLL
Mean Precision	0.5805
Mean Recall	0.5894
Mean F1-Score	0.5756

Table 2: GLiNER NER test results

Despite GLiNER being the state-of-the-art model for named entity recognition claimed to achieve near-perfect scores (Zaratiana et al. 2023), the results in Table 2 suggest otherwise. This, however, does not stem from weaknesses of the model but from annotation errors and incompleteness in the datasets, as well as disagreements between the model and the dataset as to what counts as an entity. Below are examples that explain how a state-of-the-art model can achieve such low results on the CoNLL dataset:

Example 1

Sentence: He told Reuters the reason was his own front-page editorial, entitled "A Chronic Mental Illness" in which he attacks compliant Arab leaders for serving U.S. and Israeli interests.

True Annotation: [('Reuters', 'organization'), ('Arab', 'miscellaneous'), ('U.S.', 'location'), ('Israeli', 'miscellaneous')]

Predicted Annotation: [('He', 'person'), ('Reuters', 'organization'), ('he', 'person'), ('U.S.', 'location')]

Precision: 0.5; Recall: 0.5; F1-score: 0.5

In this example, GLiNER classifies the pronoun ‘He’ as an entity, but does not classify ‘Arab’ and ‘Israeli’ as entities, contrary to the solution provided by the dataset. Indeed, ‘Arab’ and ‘Israeli’ should not be labeled as entities as they are adjectives and not entities of their own.

Example 2

Sentence: The committee consists of the chiefs of defence staff of each alliance country except Iceland, which has no armed forces.

True Annotation: [('Iceland', 'location')]

Predicted Annotation: [('The committee', 'organization'), ('chiefs of defence staff', 'person'), ('Iceland', 'location')]

Precision: 0.3333333333333333; Recall: 1.0; F1-score: 0.5

In this example, GLiNER finds additional entities: ‘The committee’ and ‘chiefs of defence staff’, which are valid entities that are unfortunately not labeled in the CoNLL dataset.

This NER analysis focuses on the CoNLL dataset because the other datasets are unsuitable: GLiNER cannot handle the sheer amount of labels of the Pile-NER dataset, and the FIGER dataset only features one labeled entity per sentence, even though there are other valid entities present, thus resulting in very low precision values as GLiNER finds those additional unmarked entities.

This matters to NEC, the main focus of this work, because we implemented NEC with GLiNER based on NER (see Section 3.1). The problem of not finding the intended entities transfers to and worsens the GLiNER NEC test results in Table 3.

5.2 NEC Overview

Model	CoNLL	FIGER-coarse	FIGER-fine
T5 NLI - zero-shot	31.07%	54.00%	21.00%
T5 MLM label - zero-shot	11.86%	72.00%	15.00%
T5 MLM entity - zero-shot	50.28%	50.00%	20.00%
T5 NLI - fine-tuned	61.02%	96.00%	53.00%
T5 MLM label - fine-tuned	58.76%	94.00%	37.00%
T5 MLM entity - fine-tuned	40.68%	56.00%	18.00%
Llama-3.1-8B prompting	72.88%	87.00%	67.00%
DeepSeek-R1 prompting	84.18%	94.00%	72.00%
Word2Vec (baseline)	23.73%	33.00%	9.00%
GLiNER NEC	61.58%	58.00%	40.00%

Table 3: Accuracy of all implemented NEC approaches in percent.

The table above was created by applying the models to 100 test instances from each dataset. The CoNLL test instances can contain multiple entities, hence the resulting percentages which do not end with .00%.

5.2.1 T5 zero-shot

All-in-all our custom task formulations for the T5 model have worked quite well, mostly performing significantly better than the baseline thus proving that this approach for NEC works.

On the FIGER-coarse dataset, our approaches are surprisingly quite competitive compared to the existing GLiNER model, with ‘T5 NLI’ and ‘T5 MLM entity’ only falling a couple of percent short of it and ‘T5 MLM label’ even beating GLiNER NEC by about 14%. On the CoNLL and FIGER-fine dataset however, our approaches fall significantly behind GLiNER. It is worth noting that while the ‘MLM label’ approach was the most performant on FIGER-coarse it was also the least performant on both CoNLL and FIGER-fine, suggesting that while being a great approach under ideal conditions it might also be the most susceptible against the problems described below.

For CoNLL the reason for the poor performance of the NLI and MLM label approaches might be that the dataset contains a label called ‘Miscellaneous’ which doesn’t work well with our task formulations. Not only is the hypothesis ‘[entity] is a Miscellaneous’ (used in both NLI and MLM label) ungrammatical but it also is semantically a lot less sensible than for the other labels. One might think that this would result in the model rarely assigning the ‘Miscellaneous’ label to a given input however we observed the opposite with the T5 MLM label approach categorizing every input as ‘Miscellaneous’ 100% of the time. This explains the low accuracy of 11.86% on CoNLL of this approach since with this behavior the model can only be correct if the gold label for the input is ‘Miscellaneous’ as well. This is significantly worse than even the Word2Vec baseline, and the NLI approach can also beat the baseline by only 7%. Since the MLM entity approach relies on representatives instead of the label name it is not affected by this and thus it performs quite well on CoNLL with 50.28% accuracy.

For FIGER-fine the reason for the poor performance is probably the high granularity and the resulting high amount of labels in the dataset. All our approaches rely heavily on the labels being sufficiently distinct from each other. In FIGER-fine however, there are many sets of labels where all of them could apply to a given entity, for example, ‘government’ and ‘government agency’, or ‘broadcast network’, ‘broadcast program’, and ‘tv channel’. While all our approaches still beat the baseline here they only do so by 6-12% and they fall significantly behind GLiNER.

5.2.2 T5 fine-tuned

When fine-tuning the T5 model for our three T5-based methods we initially observe a decrease of the evaluation loss as expected. However, rather quickly after less than 20 epochs, we observed a stark increase of the evaluation loss on all models. See Figure 1, Figure 2 and Figure 3 for the respective loss curves.

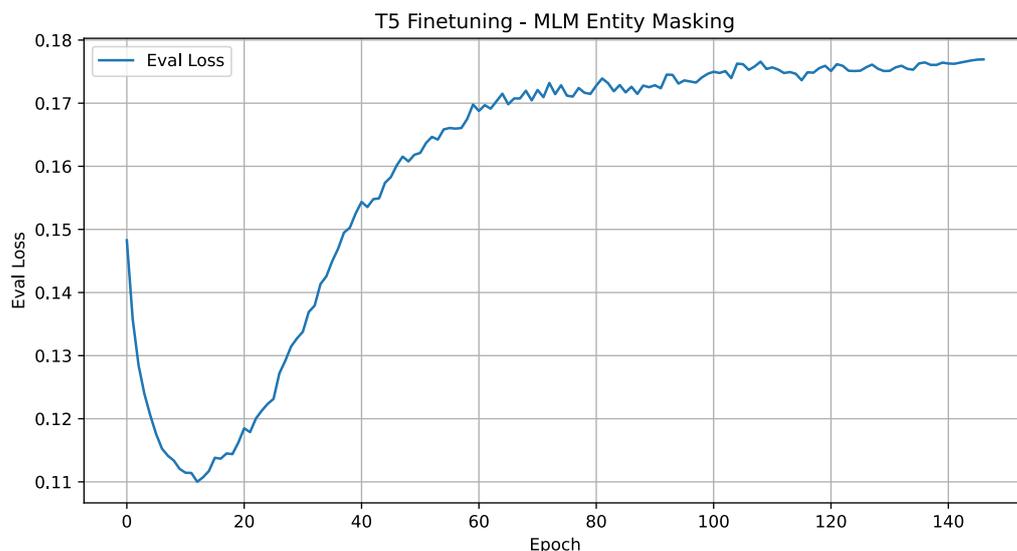


Figure 1: T5 Fine-tuning loss curve for MLM (entity masking)

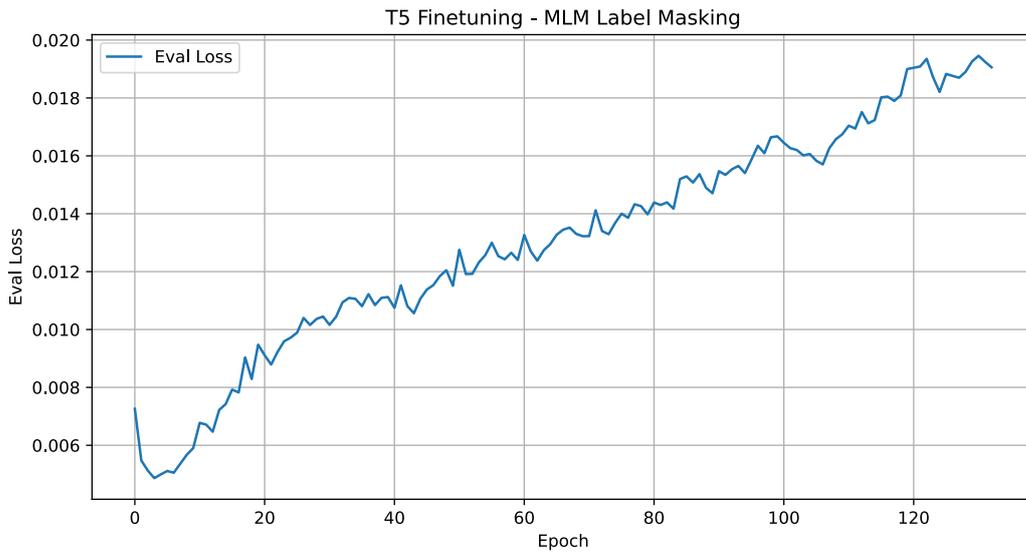


Figure 2: T5 Fine-tuning loss curve for MLM (label masking)

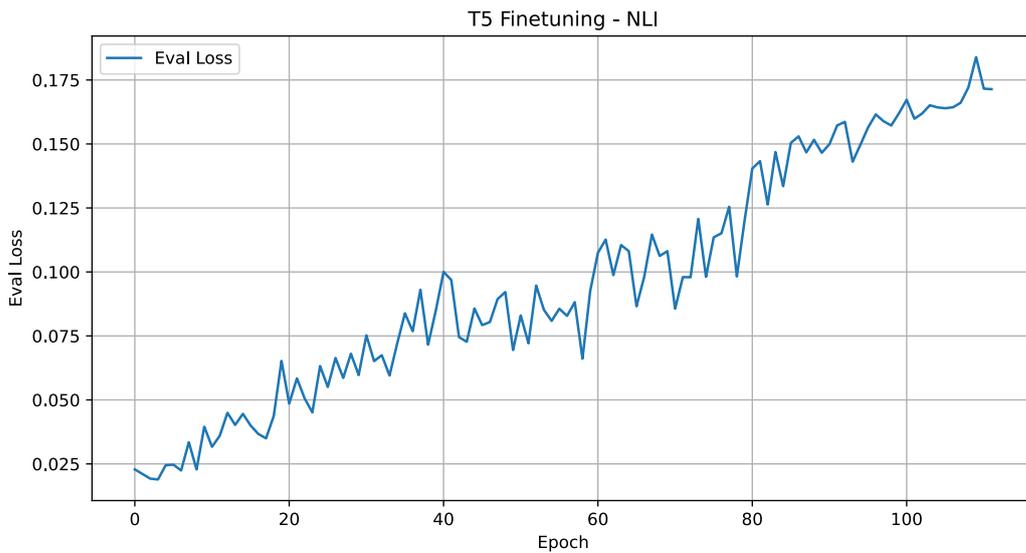


Figure 3: T5 Fine-tuning loss curve for NLI

We attribute this behavior to over-fitting. As the number of training samples we use is fairly small at several thousand, the model is easily able to memorize our training samples instead of generalizing. We therefore evaluate the fine-tuned models both at the lowest point in their loss curve, as well as at the 100 epoch point, and compare the overall prediction accuracy of the NEC task between those two points.

As expected, we observe that the prediction accuracy is highest at the lowest point of the loss curve for almost all experiments. Table 3 therefore shows these better results. For reference, we include the results after the full 100 epochs of training in the appendix under Section A.2.

Overall, fine-tuning leads to consistent improvements across the NLI and MLM label model and all datasets. For the MLM entity model fine-tuning results in slightly worse performance for all datasets except FIGER-coarse. This is likely due to only being able to pick one of many possible representatives per example during fine-tuning.

Even though we only fine-tune using data from the FIGER-coarse dataset, major improvements can be seen across all datasets. This is likely due to the use of a very generic T5 model for our zero-shot testing so any amount of fine-tuning with a comparable task setting will likely improve the models’ performance. Surprisingly, the NLI-based approach and the MLM label masking approach are able to achieve near-perfect accuracies on the FIGER-coarse dataset. This is despite ensuring no overlap between fine-tuning and evaluation datasets. Over-fitting is therefore not a factor in our evaluation performance.

5.2.3 LLM prompting

The following subsections show and interpret the results for both the named entity recognition and named entity classification methods for our LLM-based approaches.

5.2.3.1 NER

	Llama-3.1-8B	DeepSeek-R1
Mean Precision	34,41%	49,65%
Mean Recall	44,63%	59,65%
Mean F1-Score	36,63%	52,06%

Table 4: LLM NER test results CoNLL

Compared to the NER results in this table, the NEC results in Table 3 appear significantly better. This may be because NER is more complex than NEC, requiring the models to first find the correct entities. The main issue is though that we have the same problems detailed in Section 5.1, finding too many/few entities due to unintuitive or plainly wrong dataset labeling. Indeed, the DeepSeek-R1 model even achieves results comparable to our SOTA NER model GLiNER in Table 2, suggesting that it is quite suitable for the NER task too.

5.2.3.2 NEC

Table 3 showcases the superiority of large language models when it comes to generalizing to new tasks. Our hypothesis of LLMs being highly competitive in NER tasks is therefore confirmed. Nevertheless, a disadvantage of large language models is that, as hypothesized in Section 3.7, the accuracy appears to drop significantly with the growing number of labels. This is seemingly confirmed by Table 3, specifically by the drop in accuracy from 94% to 72% from the FIGER-coarse to the FIGER-fine dataset of the DeepSeek-R1 model, the only difference being the number of labels. However, upon closer inspection and manually checking the logs from the test runs for DeepSeek-R1, the instances evaluated as false are mostly due to errors or ambiguity in the datasets,

which drastically increase with the number of labels. We analyze this in more detail in Section 5.3. Table 5 shows that the accuracy does decrease with the number of labels, but not as much as the initial results suggest. These results support the concept of large language models being a useful universal tool for language tasks. Unsurprisingly, the DeepSeek-R1 reasoning model performed better than the small Llama-3.1-8B model on all datasets. Its reasoning and self-checking capabilities enhance the reliability of its predictions, but make it slow and expensive compared to other methods.

5.3 Dataset Reliability

To analyze the reliability of the datasets we sift through the generated logs of our best-performing model, DeepSeek-R1, and investigate the cause of the errors, instances where the predicted label is not the true label provided by the dataset.

5.3.1 CoNLL

Since there can be multiple entities in CoNLL sentences, we have 177 classified entities for our test run of 100 sentences. We found 20 ambiguous instances where the label predicted by DeepSeek-R1 is correct but not equal to the true label, this is mostly due to instances that are not even sentences, see Section A.3.1 for an example.

A common theme regarding the ambiguous instances is the location/organization ambiguity, for example a country can be referred to as a location as well as an organization.

We also found 5 non-entity adjectives labeled as ‘miscellaneous’, to which DeepSeek-R1 then assigns the corresponding label of the subject. For example, ‘Syrian’ in ‘Syrian President’ is labeled as ‘miscellaneous’, and DeepSeek-R1 assigns it the label ‘person’.

We discovered only two actual mis-predictions of the DeepSeek-R1 model which can be found in Section A.3.1.

Therefore, the DeepSeek-R1 model got 175 out of 177 instances in the GLiNER test run correct (disregarding the possibility of it mis-predicting an entity that is also mislabeled in the dataset with the same label the model incorrectly predicted) and thus achieved an accuracy of 98.87%. Accordingly, to account for the dataset problems, all CoNLL results can be multiplied by (0.9887/0.8418), or increased by a factor of 1.175. Table 5 contains the with this technique compensated values.

5.3.2 FIGER-coarse

This dataset appears to be more reliable than CoNLL. The instances are all valid sentences and DeepSeek-R1 thus achieved 94% accuracy, leaving us with six predictions evaluated as incorrect to analyze.

Three instances were due to ambiguity and in two instances, the target entity was not in the sentence. In these cases, the true label is still more intuitive than the model prediction, hence we shall count these as model errors anyway. Example instances as well as one wrong model prediction can be found in Section A.3.2.

Thus, DeepSeek’s true accuracy on the test sample from the FIGER-coarse dataset is 97%. To compensate for the dataset errors, the values in the FIGER-coarse column in Table 3 should be increased by a factor of 1.043.

5.3.3 FIGER-fine

On the sample from the FIGER-fine dataset, DeepSeek-R1 achieved 72% accuracy, which means there are 28 mistakes to investigate. Ambiguity and annotation errors are expected to be significantly more common compared to FIGER-coarse due to the increase in labels. There will be more instances where multiple labels fit to an entity, due to label overlap, e.g. artist vs. musician. Not all entities have all the labels that would fit them assigned, therefore we expect there to be numerous instances where sensible predictions are evaluated as false. Indeed, in the sample of 100 instances, we found 9 such ambiguities, examples of which can be found in Section A.3.3.

Somewhat surprisingly, the number of labels that are not incomplete or ambiguous, but plainly wrong is even higher, we discovered 11 such occurrences.

We expected our model to perform worse on this dataset due to the increased number of labels, and indeed we discovered 7 instances where our model arguably made an error. For this there can be different reasons, the solution not being deductible from the context being one of them. Examples can once again be found in Section A.3.3.

Hence the true accuracy of DeepSeek-R1 on the FIGER-fine dataset is approximately 93%. The FIGER column in Table 5 can thus be multiplied by 1.29 to compensate for the labeling errors and ambiguities.

5.3.4 Compensating for the Dataset Problems

Model	CoNLL	FIGER-coarse	FIGER-fine
T5 NLI - zero-shot	36.51%	56.32%	27.09%
T5 MLM label - zero-shot	13.94%	75.10%	19.35%
T5 MLM entity - zero-shot	59.08%	52.15%	25.80%
T5 NLI - fine-tuned	71.70%	100.13%	68.37%
T5 MLM label - fine-tuned	69.04%	98.04%	47.73%
T5 MLM entity - fine-tuned	47.80%	58.41%	23.22%
Llama-3.1-8B prompting	85.63%	90.74%	86.43%
DeepSeek-R1 prompting	98.87%	97.00%	93.00%
Word2Vec (baseline)	27.88%	34.42%	11.61%
GLiNER NEC	72.36%	60.49%	51.60%

Table 5: Compensated Accuracies

Note that while this table should reflect the true performance of the tested approaches better than Table 3, these results are still subject to large uncertainty due to the small sample size of 100 test instances. Our crude method of scaling the values by a factor

calculated from the ratio of the original and manually corrected accuracy of one model caused the accuracy of T5 on FIGER-coarse to be slightly larger than 100%, which of course is not really possible.

5.4 Context Relevance

In the following we will try to gain insight into what information is most important to classify the entity correctly.

5.4.1 Relevance of specific words in the input sentence

We perform the experiment described in Section 4.2 on the fine-tuned NLI model in order to determine which parts of the context are most relevant in accurately predicting the correct class. In the following, we will look at some hand-picked example sentences, where masking out any of the words marked in red will result in a mis-prediction. The entity to be classified is highlighted in bold text:

Example 1

The **1923 Stanley Cup Final** was contested by the NHL champion **Ottawa Senators** and the **WCHL** champion **Edmonton Eskimos**.

Entity: Edmonton Eskimos, Correct label: *organization*

When replacing any of the red words with *[blank]* the model will predict the label *person* instead.

In this example, context knowledge about the Stanley Cup is required to know that it is a contest between teams and not between individuals. Without this knowledge, ‘Ottawa Senators’, and ‘Edmonton Eskimos’ may just as well be unconventional names. Thus, the prediction is close enough that seemingly random disturbances lead to a mis-prediction in this setting.

Example 2

Bard is a town and comune in the **Aosta Valley region of northwestern Italy**.

Entity: Italy, Correct label: *location*

When replacing any of the red words with *[blank]* the model will predict the label *organization* instead.

From our datasets, we see that countries are often a special case as they can act as both a *location* and an *organization* depending on the context in which they are found. In this case the *location* label is correct. However, just like in the previous example, slight disturbances in the input can result in mis-prediction.

Example 3

He returned to Club Africain shortly afterwards, and is now Head Coach of **Jendouba Sport** in Tunisia, recently gaining promotion to Ligue 1.

Entity: Jendouba Sport, Correct label: *organization*

When replacing any of the red words with *[blank]* the model will predict the label *person* instead.

This example is interesting as it can be explained intuitively. When the preposition ‘of’ is missing, the relation between ‘Head Coach’ and ‘Jendouba Sport’ is unclear. It may therefore be interpreted as ‘Head Coach Jendouba Sport’, where ‘Jendouba Sport’ is the name of the coach, thus leading to a mis-prediction as *person*.

Example 4

In Canada, the term Commander-in-Chief of the Canadian Forces (in French: Commandant en chef des Forces canadiennes) can refer to both the position of supreme commander of the country ‘s armed forces and to the title granted to the viceroy.

Entity: Canada, Correct label: *location*

When replacing any of the red words with *[blank]* the model will predict the label *organization* instead.

This is another example of the country ambiguity referenced in Example 2. In this case word ‘In’ is important to identify Canada as a place rather than a political/military actor in this sentence’s context.

5.4.2 Relevance of the surrounding context compared to the entity itself

The ‘T5 MLM entity’ has the entity itself masked thus only having the context of the entity as input. While it outperforms the two other models on CoNLL for probably unrelated reasons already described in Section 5.2.1 it falls behind on FIGER-coarse by 13% and is quite similar to the other two T5 approaches on FIGER-coarse. This suggests that while the entity itself does play a role, the context of the entity might be much more important to the T5 model, at least if presented with class representatives instead of the actual entity.

The Word2Vec model uses only the entity itself as input for its classification, having no information about the surrounding context. While its achieved accuracy is significantly higher than random guessing it is not able to compete with any of the other approaches we explored. This is in part due to the high number of multi-token entities, for which no embeddings exist and which are therefore mis-predicted. Additionally, the improvement of the Word2Vec approach when compared to random guessing is higher for the more fine-grained FIGER-fine dataset. We hypothesize that this is due to the representatives within each class being closer to each other in the embedding space when using more fine-grained labels.

These results confirm that the entity itself carries sufficient information to enable correct classification in some cases. The large gap in the accuracies achieved with the Word2Vec approach when compared with our other approaches can be explained as a combination

of the missing context and the previously mentioned drawbacks of the approach like lack of support for multi-word entities and missing embeddings.

6 Discussion and Outlook

All in all, we learned about both the effectiveness of our different NER and NEC approaches and the evaluation methods used to test them.

6.1 NLP T5 task formulations

We can conclude that formulating NEC as a different NLP task for the T5 model does indeed work and yields good results, especially after fine-tuning, with the NLI formulation being the most promising one. Depending on the dataset and label set used the MLM task formulations also show some merit, most notably in a zero-shot scenario.

However, these approaches have also some serious limitations when it comes to efficiency, the size of the label set they are able to handle, and the exact nature of the label set. For each entity, they require one or more tests for each label which does not scale well for larger label sets. The labels also have to be a good fit for these task formulations, with more abstract labels (like ‘Miscellaneous’) or too similar labels being rather problematic.

Possible future work might include experiments with and fine-tuning of T5 models other than the base variant, like the large variant, the 11b variant, or variants that are already more specifically trained for the tasks we would use them for (e.g. the XXL true NLI variant of T5).

Another interesting future experiment might be to fine-tune T5 on different datasets and measure how this fine-tuning transfers to different datasets that use different label sets. In this report, we did all our fine-tuning on the FIGER-coarse dataset and then evaluated the resulting models on all three datasets. It might prove interesting to have different models fine-tuned on CoNLL and FIGER-fine to find out which dataset is best suited for training given that the model is evaluated on a different dataset with a different set of labels.

6.2 Regarding our hypotheses

We were able to successfully confirm or deny all of our three hypotheses described in Section 4.3.

While fine-tuning resulted in major improvements in model performance we also saw some good accuracies for our two MLM approaches on some datasets in a zero-shot setting. Although it may not be a strict requirement, fine-tuning still resulted in much-improved numbers making it, depending on the expectations, relatively important for our custom task formulations.

When considering that FIGER-fine has many more labels than the other datasets making it a lot harder to perform well by chance Word2Vec indeed performed better on the more fine-grained dataset relatively speaking, while the T5 models much preferred a more coarse granularity as we predicted.

Contrary to our expectations the fine-tuning performed on FIGER-coarse resulted in sizeable performance improvements on the other datasets which use different label sets

as well. While the fine-tuned models still performed best when using the same label set as they were fine-tuned with, they were definitely able to transfer their knowledge gained from fine-tuning onto different label sets.

6.3 Competitiveness of Large Language Models in NER tasks

We found that LLMs are very good at NER and NEC tasks, with DeepSeek-R1 having the overall best performance compared to all our approaches, without any required fine-tuning. While LLMs provide a very good performance they are however less efficient than more specialized models. Especially the use of reasoning models is rather excessive for this task.

We hypothesize that a larger non-reasoning language model such as GPT-4o or DeepSeek-V3 (DeepSeek-AI 2024) can achieve similar results with much quicker response times since it does not have to output the entire thought process behind each predicted label.

Our results are a demonstration of the suitability of LLMs as a general solution for NLP tasks.

6.4 Poor dataset quality

We observed that the datasets we used often include oddities like cut-off sentences and often mislabel entities as well. We have mentioned these reliability issues in Section 2.2.1, Section 2.2.3 and Section 5.3. This heavily impacts our accuracy metrics since even a perfect NER model would not be able to achieve 100% accuracy on an erroneous dataset. Possible solutions would be to either search for different more reliable datasets or to manually annotate CoNLL and FIGER. The latter would be feasible at least for our evaluation sets which for each dataset only consists of 100 instances.

Bibliography

- DeepSeek-AI. 2024. “Deepseek-V3 Technical Report”. 2024. <https://arxiv.org/abs/2412.19437>.
- DeepSeek-AI. 2025. “Deepseek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning”. 2025. <https://arxiv.org/abs/2501.12948>.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding”. *Corr.* <http://arxiv.org/abs/1810.04805>.
- Grattafiori, Aaron, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, et al. 2024. “The Llama 3 Herd of Models”. 2024. <https://arxiv.org/abs/2407.21783>.
- Honovich, Or, Roei Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansky, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. “TRUE: Re-Evaluating Factual Consistency Evaluation”. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 3905–20. Seattle, United States: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2022.naacl-main.287>.
- Ling, Xiao, and Daniel Weld. 2021. “Fine-Grained Entity Recognition”. *Proceedings of the AAAI Conference on Artificial Intelligence* 26 (1): 94–100. <https://doi.org/10.1609/aaai.v26i1.8122>.
- Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. “Efficient Estimation of Word Representations in Vector Space”. 2013. <https://arxiv.org/abs/1301.3781>.
- Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. 2023. <https://arxiv.org/abs/1910.10683>.
- Rücker, Susanna, and Alan Akbik. 2023. “CleanCoNLL: A Nearly Noise-Free Named Entity Recognition Dataset”. 2023. <https://arxiv.org/abs/2310.16225>.
- Tjong Kim Sang, Erik F., and Fien De Meulder. 2003. “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 142–47. <https://aclanthology.org/W03-0419/>.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. “Attention Is All You Need”. 2023. <https://arxiv.org/abs/1706.03762>.

Zaratiana, Urchade, Nadi Tomeh, Pierre Holat, and Thierry Charnois. 2023. “Gliner: Generalist Model for Named Entity Recognition Using Bidirectional Transformer”. 2023. <https://arxiv.org/abs/2311.08526>.

Zhou, Wenxuan, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. 2023. “Universalner: Targeted Distillation from Large Language Models for Open Named Entity Recognition”. <https://arxiv.org/abs/2308.03279>.

Individual Author Contributions

Please refer to the commit history of the project's Git repository (<https://gitlab.cl.uni-heidelberg.de/fsem25-project-nerds/ner-project/>) for viewing the individual contributions of each of the authors to this project.

The authors may appear under one of the following names in the Git commit history:

- Julian Partanen: JulianFP
- Paul Kupper: Paul Kupper, kupper, ih322@uni-heidelberg.de
- Thomas Wolf: Thomas Wolf

This project report was written collaboratively, with each author primarily focusing on the tasks they implemented in code.

A Appendix

A.1 FIGER - list of all labels

This is a list of all labels used in our FIGER dataset evaluations (excluding quotations marks and commas):

FIGER-coarse: ‘person’, ‘organization’, ‘location’, ‘product’, ‘art’, ‘event’, ‘building’

FIGER-fine: ‘actor’, ‘architect’, ‘artist’, ‘athlete’, ‘author’, ‘coach’, ‘director’, ‘doctor’, ‘engineer’, ‘monarch’, ‘musician’, ‘politician’, ‘religious leader’, ‘soldier’, ‘terrorist’, ‘airline’, ‘company’, ‘educational institution’, ‘fraternity sorority’, ‘sports league’, ‘sports team’, ‘terrorist organization’, ‘government agency’, ‘government’, ‘political party’, ‘educational department’, ‘military’, ‘news agency’, ‘city’, ‘country’, ‘county’, ‘province’, ‘railway’, ‘road’, ‘bridge’, ‘body of water’, ‘island’, ‘mountain’, ‘glacier’, ‘astral body’, ‘cemetery’, ‘park’, ‘engine’, ‘airplane’, ‘car’, ‘ship’, ‘spacecraft’, ‘train’, ‘camera’, ‘mobile phone’, ‘computer’, ‘software’, ‘game’, ‘instrument’, ‘weapon’, ‘film’, ‘play’, ‘written work’, ‘newspaper’, ‘music’, ‘attack’, ‘election’, ‘protest’, ‘military conflict’, ‘natural disaster’, ‘sports event’, ‘terrorist attack’, ‘airport’, ‘dam’, ‘hospital’, ‘hotel’, ‘library’, ‘power station’, ‘restaurant’, ‘sports facility’, ‘theater’, ‘point in time’, ‘color’, ‘award’, ‘educational degree’, ‘title’, ‘law’, ‘ethnicity’, ‘language’, ‘religion’, ‘god’, ‘chemical thing’, ‘biological thing’, ‘medical treatment’, ‘disease’, ‘symptom’, ‘drug’, ‘body part’, ‘living thing’, ‘animal’, ‘food’, ‘website’, ‘broadcast network’, ‘broadcast program’, ‘tv channel’, ‘currency’, ‘stock exchange’, ‘algorithm’, ‘programming language’, ‘transit system’, ‘transit line’

A.2 Accuracy after 100 Epochs of Fine-tuning

Table Table 6 shows the non-corrected accuracies when using the model variants trained for the full 100 epochs. For almost all test cases we observe significantly worse accuracies than when stopping at the lowest point in the loss curve. The two exceptions occur with the T5 MLM entity model where there are slight improvements in the accuracy on the CoNLL and FIGER-coarse datasets.

Model	CoNLL	FIGER-coarse	FIGER-fine
T5 NLI - fine-tuned	54.52%	61.48%	22.32%
T5 MLM label - fine-tuned	36.16%	74.31%	15.00%
T5 MLM entity - fine-tuned	41.81%	60.00%	16.00%

Table 6: Accuracy of fine-tuned models after 100 epochs in percent.

A.3 Examples from the Dataset Reliability Section

These examples and more can be found in the NEC evaluation logs in our git project.

A.3.1 GLiNER

Incomplete sentence instance example:

Instance 23:

Sentence: 52) v Worcestershire.

Entity: Worcestershire

True Label(s): organization

Predicted Label: location

DeepSeek-R1 predictions we ruled as false:

Instance 16:

Sentence: In Bistrita: Gloria Bistrita(Romania) 2 Valletta(Malta) 1

Entity: Gloria Bistrita

True Label(s): organization

Predicted Label: person

Instance 55:

Sentence: Prime Minister Benjamin Netanyahu 's government, which took office in June, has said it will not allow the Authority, set up under a 1993 interim peace deal to control parts of the Gaza Strip and West Bank, to operate in Jerusalem.

Entity: Authority

True Label(s): organization

Predicted Label: miscellaneous

A.3.2 FIGER-coarse

Ambiguity example:

Instance 46:

Sentence: He cites his musical influences in metal as: Manowar, Iron Maiden, Gamma Ray, Angra, Helloween, Running Wild, Metallica, Megadeth, Judas Priest, Nightwish, Spinal Tap, Iced Earth, Dio.

Entity: Dio

True Label(s): person

Predicted Label: organization

Target entity not in sentence example:

Instance 19:

Sentence: To some extent he agreed with Fritz Fischer 's assessment that the differences between Imperial, Weimar and Nazi foreign policy were of degree rather than kind.

Entity: Nazi Germany

True Label(s): location

Predicted Label: event

False model prediction example:

Instance 38:

Sentence: The 1923 Stanley Cup Final was contested by the NHL champion Ottawa Senators and the WCHL champion Edmonton Eskimos.

Entity: Edmonton Eskimos

True Label(s): organization

Predicted Label: event

A.3.3 FIGER-fine

Ambiguity examples:

Instance 4:

Sentence: The family seat was Bantry House, near Bantry, in County Cork, Ireland.

Entity: Bantry

True Label(s): city

Predicted Label: town

Instance 39:

Sentence: It stayed there for two weeks, before being dethroned by Boyz II Men 's " End of the Road " .

Entity: End of the Road

True Label(s): music

Predicted Label: songs

Wrong labeling example:

Instance 21:

Sentence: Barrow was born in Wolfskin District, Oglethorpe County, Georgia on October 18, 1852.

Entity: Georgia

True Label(s): province

Predicted Label: state

Georgia joined the United States in 1788. The U.S. does not have provinces — only states and territories.

False model predictions examples:

Instance 16:

Sentence: Holger Apfel (born December 29, 1970 in Hildesheim, Lower Saxony) is the leader of the right-extremist National Democratic Party of Germany in Saxony and has served as a member of the Saxon Parliament since 2004.

Entity: Landtag of Saxony

True Label(s): government

Predicted Label: government agency

Instance 63:

Sentence: " Veronica Sawyer " (originally by Edna 's Goldfish) featured on Fame, Fortune and Fornication.

Entity: Fame, Fortune and Fornication

True Label(s): music

Predicted Label: broadcast program

Instance 90:

Sentence: There are approximately eight hundred Gyu-Kaku locations within Japan,

and locations have also been opened in the United States (New York City, California, Hawaii), Hong Kong, Taipei, Indonesia and Singapore.

Entity: Indonesia

True Label(s): country, government

Predicted Label: predicted_class