Looking Right is Sometimes Right: Investigating the Capabilities of Decoder-only LLMs for Sequence Labeling

David Dukić[†] **Jan Šnajder**
TakeLab, Faculty of Electrical Engineering and Computing, University of Zagreb
`{david.dukic, jan.snajder}@fer.hr`

Abstract

Pre-trained language models based on masked language modeling (MLM) excel in natural language understanding (NLU) tasks. While fine-tuned MLM-based encoders consistently outperform causal language modeling decoders of comparable size, recent decoder-only large language models (LLMs) perform on par with smaller MLM-based encoders. Although their performance improves with scale, LLMs fall short of achieving state-of-the-art results in information extraction (IE) tasks, many of which are formulated as sequence labeling (SL). We hypothesize that LLMs' poor SL performance stems from causal masking, which prevents the model from attending to tokens on the right of the current token. Yet, how exactly and to what extent LLMs' performance on SL can be improved remains unclear. We explore techniques for improving the SL performance of open LLMs on IE tasks by applying layer-wise removal of the causal mask (CM) during LLM fine-tuning. This approach yields performance gains competitive with state-of-the-art SL models, matching or outperforming the results of CM removal from all blocks. Our findings hold for diverse SL tasks, demonstrating that open LLMs with layer-dependent CM removal outperform strong MLM-based encoders and even instruction-tuned LLMs.[1]

1 Introduction

Pre-trained language models (PLMs) built upon the Transformer architecture have demonstrated exceptional performance across many natural language understanding (NLU) tasks (Radford et al., 2018; Devlin et al., 2019; Yang et al., 2019; Clark et al., 2020; Raffel et al., 2020). Typically, achieving state-of-the-art (SOTA) results in tasks such as sequence classification and sequence labeling involves a two-step process: pre-training on unlabeled corpora, followed by fine-tuning on task-specific data – a process often referred to as transfer learning (Ruder et al., 2019; Raffel et al., 2020). Two prevailing architectures emerged, each coupled with a compatible pre-training paradigm: (1) the decoder-only architecture, utilizing causal language modeling (CLM) for pre-training, and (2) the encoder-only architecture, with the masked language modeling (MLM) pre-training objective.

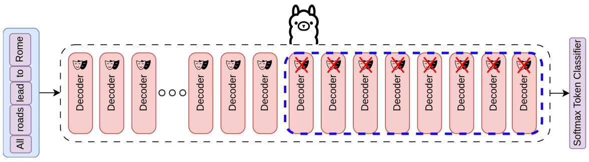


Figure 1: Layer-wise causal mask removal from decoder block groups in a decoder-only LLM. Here, the causal mask is removed from the top eight decoder blocks of the Llama2-7B model to enable bidirectionality during fine-tuning, which proves beneficial for many SL tasks.

In transfer learning experiments that juxtapose models of a comparable number of parameters, MLM-based encoders consistently outperformed CLM-based decoders on NLU tasks (Devlin et al., 2019).[2] However, a shift in strategy emerged within the NLP community when encoder models ceased being scaled up to the same magnitude of parameters and pre-training data as their decoder counterparts. Consequently, there has been a pronounced trend toward scaling decoder models to multiple billion parameters, leading to a proliferation of large language models (LLMs). Combining LLM text generation capabilities with various prompting strategies can boost the performance on many NLU tasks, eliminating the need for fine-tuning model parameters (Liu et al., 2023).

Despite LLMs' good performance on NLU tasks,

[†] Corresponding author: `david.dukic@fer.hr`

[1] Code: `https://github.com/dd1497/llm-unmasking`.

[2] Henceforth, we use the terms "encoder" and "decoder" to refer exclusively to MLM- and CLM-based encoder- and decoder-only variants, respectively.

Findings of the Association for Computational Linguistics: ACL 2024, pages 14168–14181
August 11-16, 2024 ©2024 Association for Computational Linguistics

there is still considerable room for improvement, even for the largest decoder models, such as ChatGPT, which fall far behind SOTA results on fundamental NLP tasks. This holds particularly for information extraction (IE) tasks (Han et al., 2023), such as named entity recognition (NER), aspect-based sentiment analysis (ABSA), and event extraction (EE). These tasks are often formulated as sequence labeling (SL), and tackling SL by prompting LLMs proved quite difficult (Wang et al., 2023a). However, it is unclear how exactly and to what extent the subpar performance of LLMs on SL tasks can be remedied.[3] As the community stopped scaling up encoders, LLMs are solidifying their position as the field's de facto standard. Thus, improving LLMs' SL performance is the sole viable option.

Although many SOTA LLMs are accessible only through paywalls, the community has responded by training and publicly releasing open LLMs with multiple billions of parameters, such as Llama2 (Touvron et al., 2023), Mistral (Jiang et al., 2023), and OPT (Zhang et al., 2022). Parameter-efficient fine-tuning (PEFT) techniques and quantization, including QLoRA (Dettmers et al., 2023), facilitate experimentation with fine-tuning these models on task-specific data. These strategies allow leveraging open LLMs' hidden states for task-specific classifiers, enabling fine-tuning on consumer-grade hardware with comparable or better performance than full fine-tuning.

The poor performance of LLMs on SL tasks can be traced back to the information encoded in the hidden states. Using the hidden state of the decoder's last token serves as a reliable feature for classifying the input sequence. However, the same tactic does not suffice for SL because the causal mask (CM), integral to endowing decoders with text generation capabilities, limits bidirectional information flow, preventing the model from "looking right," i.e., attending to tokens in the sequence positioned to the right of the current token. For many SL tasks, the token's label depends on the succeeding tokens, and omitting this context often results in subpar performance. Recently, Li et al. (2023) found that completely removing the CM from the Llama2 model during fine-tuning substantially enhances SL performance on NER. Others also experimented with the CM removal and enabling bidirectionality concurrently with our work (BehnamGhader et al., 2024; Lee et al., 2024).

In this paper, we explore techniques to enhance the SL performance of open LLMs on IE tasks. Building on insight from Li et al. (2023), we hypothesize that removing the CM from all decoder layers may not be beneficial for all SL tasks. To confirm this, we experiment with layer-wise CM removal across LLM blocks, observing gains competitive with SOTA models, in contrast to removing or keeping the CM in all layers. We extend our analysis to a series of IE tasks (NER, ABSA, and EE) and demonstrate that layer-wise "looking right" is highly task-dependent. We compare against strong encoder-only sequence taggers and instruction-tuned LLMs. Open LLMs without CMs in specific layers outperform these baselines in almost all scenarios. Finally, we compare with encoder-only models, assessing the relative strengths of the decoder and encoder architectures regarding parameter scale, training data, and the particular SL task. We pre-train small encoder and decoder models from scratch with an identical number of parameters on the same data. Our findings indicate the superiority of the same-scale encoder architecture over the decoder on SL tasks, which is consistent even when the CM is removed while fine-tuning the small pre-trained decoders. This suggests that the observed SL performance gain in LLMs upon CM removal stems from scaling up.

Our contributions are threefold: (1) We present evidence that layer-wise CM removal, followed by supervised fine-tuning, improves the performance of decoder-only LLMs on SL tasks compared to removing or keeping CM in all layers; (2) Layer-wise CM removal configuration that yields the highest gains strongly depends on the peculiarities of the SL task; (3) We show that the CM removal effect brings no gains on the small scale. We believe our results will contribute to building more performant decoder-only SL models.

2 Background and Related Work

Encoders and Decoders. The first encoder-only PLM, with MLM pre-training success, was BERT (Devlin et al., 2019), while the first decoder-only PLM with CLM pre-training that gained traction was GPT (Radford et al., 2018). Shoeybi et al. (2019) showed that scaling the BERT model to four billion parameters with MLM and the GPT-2

[3] Although LLMs can solve many NLU tasks, including, but not limited to, IE tasks used for the knowledge base population, they have their limitations. We still need specialized methods for SL-based IE where we want to identify and count information elements in the text. This is especially useful for computational social science applications.

model (Radford et al., 2019) to eight billion parameters with CLM brings gains on NLU tasks. Nevertheless, the community stopped scaling up encoders and continued scaling decoders. This shift in strategy has given rise to notable properties in decoder-only LLMs, including enhanced zero- and few-shot learning capabilities (Wei et al., 2021; Brown et al., 2020), the ability for in-context learning (ICL) (Brown et al., 2020; Liu et al., 2023), and chain-of-thought prompting (Wei et al., 2022). Examples of LLMs with these emerging properties are GPT-3 (Brown et al., 2020), PaLM (Chowdhery et al., 2023), and ChatGPT.

Pre-training with CLM teaches the model to generate text and does not allow it to use the sequence context right to the token being processed. This is enabled via a CM, which prevents the model from attending to future tokens. While beneficial and necessary for coherent text generation, restricting the model to look right is detrimental if one wants to use the decoder to produce fully contextualized token-level embeddings with the decoder. Although decoder-only models perform poorly on sequence *labeling* tasks (i.e., tasks where each individual token is assigned a label), they perform quite well on sequence *classification* tasks (i.e., tasks where the entire sequence is assigned one label). Therefore, many decoder models implement sequence classification heads on top of decoder blocks but not SL heads (Wolf et al., 2020).

Li et al. (2023) found that removing the CM from all decoder layers of Llama2 (Touvron et al., 2023) improves the performance of named entity recognition (NER). However, Li et al. (2023) did not examine in depth why this phenomenon occurs, whether it occurs with layer-wise CM removal, and whether it persists for other IE tasks. We build on this and inspect the effect of different unmasking configurations across layers of open decoder-only LLMs. Furthermore, we train small decoders with CLM and small encoders with MLM to demonstrate that the superiority of unmasked LLMs is a consequence of the model and training data scale.

Universal IE. Many IE tasks, such as NER, subtasks of ABSA, and EE, can effectively be formulated as SL tasks. Traditionally, these tasks were addressed with encoders, mapping each token's hidden state to task label space with a linear layer on top of the last encoder block (Pontiki et al., 2015; Zhang et al., 2020; Zhou and Chen, 2021). However, this approach necessitates training a separate model for each task. More recent methods aim to infer the underlying structure of multiple IE tasks simultaneously and develop universal information extraction (UIE) models, employing encoder, decoder, or encoder-decoder architectures (Paolini et al., 2021; Lu et al., 2022; Fei et al., 2022; Wang et al., 2022a; Ping et al., 2023; Zhu et al., 2023; Lou et al., 2023; Ding et al., 2024). While the practical advantages of UIE are obvious, including more IE tasks would still require re-training the UIE extractors. Therefore, we stick to the one model per SL task strategy, relying on parameter-efficient methods that yield compact and extensible models (Houlsby et al., 2019).

In-context Learning for IE. ICL is a powerful method for scrutinizing LLMs, employed by prompting LLMs with task demonstrations. IE tasks can be tackled with ICL, relying on the emergent properties of LLMs. Nevertheless, Pang et al. (2023) report that the performance of LLMs leveraging ICL lags behind the SOTA results of supervised IE models. They develop a model that learns to guide the prompt for improved LLM performance. Another notable example is GPT-NER (Wang et al., 2023a), transforming the NER task into a generation one utilizing ICL and ChatGPT, while Blevins et al. (2023) and Mehta et al. (2024) apply prompt-based methods to structured prediction tasks such as NER and semantic role labeling.

Instruction Tuning for IE. Releasing the weights of encoder-decoder and decoder-only pretrained LLMs paved the way for instruction tuning (IT) (Mishra et al., 2022). This paradigm involves fine-tuning the model to address specific tasks in a supervised generative manner, providing the model with instructions for each data point. SL tasks can be adapted for compatibility with IT. Arguing that current prompt templates are primarily designed for sentence-level tasks and inappropriate for SL objectives, Wang et al. (2022b) reformulate NER as a generation problem relying on improved prompt templates and IT. Drawing on the fusion of UIE and IT, Wang et al. (2023b) reveal that multi-task IT can give results comparable to BERT in a supervised setting for NER. Finally, Scaria et al. (2023) find that instruction-tuned encoder-decoder PLMs excel at ABSA subtasks, leveraging the Tk-instruct model (Wang et al., 2022c). Given that IT is a powerful supervised paradigm for combining LLMs' generation abilities with labeled data, we compare SL models against IT baselines.

3 Layer Group Unmasking

The CM is a crucial component of CLM-based decoders, preventing the model from attending to future tokens and facilitating autoregressive text generation. This constraint is enforced by defining the CM as a triangular matrix and adding this matrix to the dot product of the query and key attention matrices. The resulting sum is passed through the softmax function in the scaled dot-product attention mechanism, as introduced by Vaswani et al. (2017). Formally:

$$CM = \begin{pmatrix} 0 & -\infty & -\infty & \dots & -\infty \\ 0 & 0 & -\infty & \dots & -\infty \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix},$$

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^T + CM}{\sqrt{d_k}}\right) V.$$

Here, Q, K, and V are the query, key, and value attention matrices, respectively, and d_k is the dimension of queries and keys. Effectively, applying softmax for tokens with value $-\infty$ in the CM results in attention scores being 0.

As reported by Li et al. (2023), removing CM across all decoder layers of Llama2 during fine-tuning increases the model's performance on the NER SL task by a large margin. This is surprising because although, during pre-training, the CM mask was in place and the model was restricted from "looking right," it learned to attend to future tokens by backpropagating on the training data over a few epochs with CM removed. Building on this and to allow the LLM to "sometimes look right," we select a subset of decoder blocks for which we replace all $-\infty$ entries in the CM with zeros, effectively removing the CM (technically, we leave the positions of padding tokens to $-\infty$, but replace all others with 0). Since we group layers into groups of b blocks, we refer to these decisions as layer group unmasking configurations.

Considering that Llama2 is made up of $n = 32$ decoder blocks, and each block can either remove or keep its CM, this gives a search space of $2^n = 2^{32}$ possibilities. To reduce the search space to a manageable size, we group the n decoder blocks into $m = 4$ layer groups with $b = 8$ consecutive decoder blocks per layer group and then jointly mask or unmask all blocks in each layer group. This leaves us with $2^m = 16$ unmasking configurations per LLM and task. For ease of reference, we encode our unmasking configurations with binary four-digit codes ranging from 0000 (all four layer groups masked) to 1111 (all four layer groups unmasked), where each 0 and 1 denote a layer group that is masked or unmasked, respectively. Unmasking configurations are interpreted left to right (the first digit pertains to the layer group closest to the model's input).

Dataset	Train	Validation	Test	Total	RDRR
CoNLL03 NER	14041	3250	3453	20744	0.593
CoNLL03 Chunking					0.518
ACE05	14672	873	711	16256	0.411
Rest14	2737	304	800	3841	0.397

Table 1: Statistics for the datasets and their splits. We show the number of sentences per split, the total number of sentences, and the right-side dependency relations ratio (RDRR).

4 Experimental Setup

4.1 Sequence Labeling Tasks

We focus our experiments on IE tasks framed as SL: NER, ABSA with aspect term extraction and polarity subtasks, and the trigger classification (TC) subtask of EE. We also include text chunking (shallow parsing), which, although not an IE task, is considered a prototypical SL task. Dataset statistics are shown in Table 1. For each dataset, we calculate its *right-side dependency relations ratio* (RDRR), defined as the ratio of right-side dependency relationships to the total number of left-side and right-side relationships counted for all labeled spans in the training set. Essentially, this metric indicates the degree to which labeled spans depend on the context to their right. We use spaCy (Honnibal et al., 2020) dependency parser to obtain the dependency relations.

NER and Text Chunking. For NER and text chunking, we choose CoNLL03, a standard and widely used benchmark (Tjong Kim Sang and De Meulder, 2003). We use the version from Hugging Face Datasets (Lhoest et al., 2021) with an IOB2 sequence tagging scheme, which has a predefined train, validation, and test split.

Aspect Term Extraction and Polarity. We use the data from SemEval-2014 Task 4 and the restaurants domain (Pontiki et al., 2014) (Rest14). We merge the first two ABSA subtasks, aspect term extraction (ATE) and aspect term polarity (ATP), into one SL task (ATE+ATP). We tokenize the dataset with spaCy (Honnibal et al., 2020) and match given character spans of aspect terms with token spans to obtain IOB2 tags (we discard 13 aspect terms that could not be matched in this way). Training and test split were predefined. Following prior work (Wang et al., 2021), we randomly sample 10% of the sentences for the validation set.

Trigger Classification. The ACE05 dataset (Doddington et al., 2004) is a widely used TC dataset. The TC task combines two EE tasks into a single SL task: trigger identification, i.e., finding spans of tokens constituting the event predicate and classifying them. We use the English train, validation, and test split obtained with the standard ACE pre-processing tool.[4] We use this tool to obtain sentences and tokens and create IOB2 tags.

Evaluation. We evaluate all tasks with micro F1 score on IOB2 tag predictions with strict matching using seqeval (Nakayama, 2018), where the predicted output span must exactly match the expected output span. We evaluate only the predictions on the first token of the tokenized words from the input sequence to obtain the same number of predictions as there are target IOB2 labels.

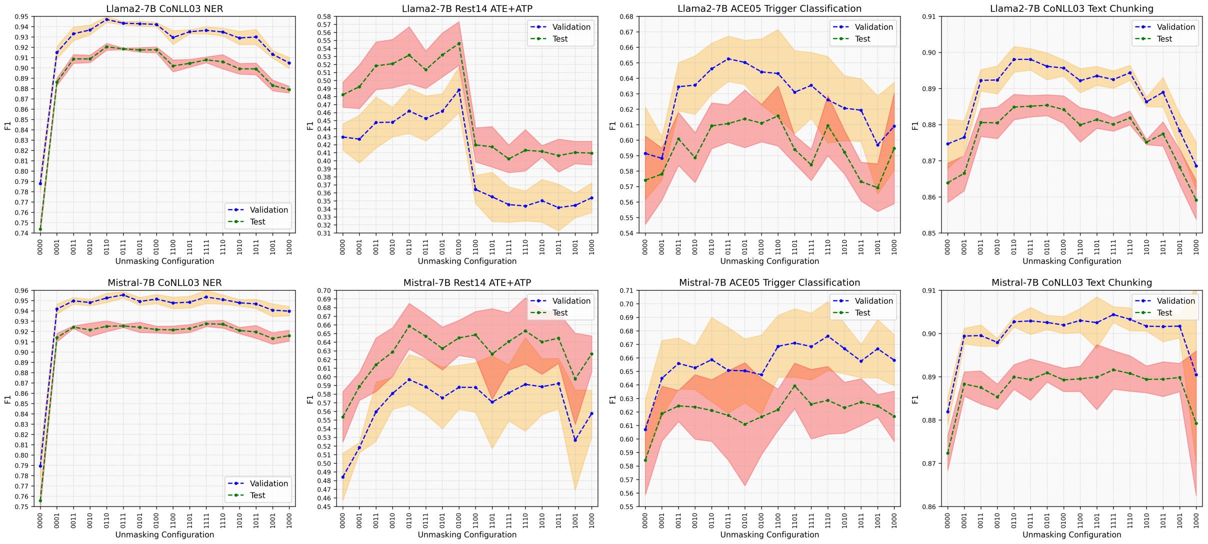
Instruction Tuning. To train LLMs using IT, each dataset needs to be further pre-processed to obtain instruction prompts. We form instructions similar to the ones used for NER by Wang et al. (2022b), although we require a more strict output response from decoder-only LLMs, in line with the output format used by Wang et al. (2023b). We create instruction prompts by parsing IOB2 tags to create desired outputs. See Appendix A.3 for instruction tuning examples. To ensure a fair evaluation consistent with models fine-tuned directly for SL, we heuristically map response spans of instruction-tuned models to IOB2 tags. We employ greedy span-based matching of predicted spans and their types with input tokens, similar to Wang et al. (2022b). We treat all cases in which no predictions are made, or all predicted spans do not align with input tokens, or an exception arises during matching due to output generation stochasticity, as if the O tag was predicted for every input token.

4.2 Training Details and Hyperparameters

Models. We choose open LLMs for unmasking experiments and IT: Llama2-7B (Touvron et al., 2023) and Mistral-7B (Jiang et al., 2023), both with 7B parameters. For encoders, we use RoBERTa-base and RoBERTa-large PLMs with 125 and 355 million parameters, respectively (Liu et al., 2019). For SL experiments, we employ a classic softmax SL head on top of the pre-trained models. Model implementations and weights are taken from Hugging Face Transformers (Wolf et al., 2020). If the SL head is not implemented, we add our implementation mimicking the one for RoBERTa models. See Appendix A.1 for more details.

Optimization and Pre-processing. We use quantized LoRA (QLoRA) (Hu et al., 2021; Dettmers et al., 2023) for fine-tuning all models to ensure a fair comparison between the smallest and the largest models and enable fine-tuning under constrained computing resources. QLoRA is applied to query and value attention matrices inside each encoder or decoder block with a fixed rank of $r = 64$, a scaling parameter of $\alpha = 16$, and a dropout probability of $p = 0.1$. This way, only decomposed query and value matrices, along with the SL head parameters, are optimized with cross-entropy loss, yielding a drastic trainable parameter reduction per model. The models are trained in bfloat16 precision, with loaded pre-trained weights in 4-bit NormalFloat data type, and we use double quantization. Using this setup, we were able to fit all models into 40GB of GPU memory of Ampere A100 and are trained with a consistent batch size of 16 per experiment. To use this batch size across models, we pre-process all datasets to a maximum tokenized sequence length of 128. This cutoff is optimal as there are <10 sentences for each dataset and split that end up truncated independent of the tokenizer used. We pad the sequences to the longest example in the batch and randomly sample examples for training depending on the seed. We train the models with paged 8-bit AdamW (Loshchilov and Hutter, 2017) optimizer to handle the memory spikes (Dettmers et al., 2023). The parameters of AdamW are fixed to $\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 1\text{e}{-5}, \lambda = 0.1$. For the learning rate scheduler, we choose cosine annealing scheduler (Loshchilov and Hutter, 2016). We apply gradient clipping set to 1.0, gradient accumulation with four steps, and gradient checkpointing. We use a consistent learning rate of $2\text{e}{-4}$ across all experiments and fine-tune models over a fixed number of five epochs. All results are averages of five runs with different seeds, and we always pick the last model for each seed.

[4] https://bit.ly/ace2005-preprocessing



(a) NER (b) ATE+ATP (c) Trigger Classification (d) Text Chunking

Figure 2: Micro F1 SL scores with decoder-based LLMs and different unmasking configurations (sorted by Gray code starting with all decoder layers masked – configuration 0000). Upper row plots show Llama2-7B model results, while lower row plots show Mistral-7B model results on validation and test sets of four SL tasks (left to right, one dataset per task). All results are averages of five runs. The shaded area corresponds to standard deviation.

Instruction Tuning Details. We use Llama2-7B and Mistral-7B with CLM heads for IT. Training is done with QLoRA based on cross-entropy loss. All hyperparameters are inherited from SL experiments except for the batch size. The batch size depends on the maximum sequence length for packing the dataset examples. Short examples are packed together in the same input to increase training efficiency. This length is set to 512 for the Rest14 dataset and 1024 for other datasets since Rest14 has shorter instruction examples on average and has the smallest training set (cf. Table 1). We observe that ATE+ATP models need double the epochs (10 epochs) to learn to extract aspect terms and their polarity, and we hypothesize that this is due to the low number of training examples compared to other datasets (~3k vs. ~14k).

5 Results

5.1 Layer Group Unmasking Results

Figure 2 shows the validation and test evaluation sets performance of Llama2-7B and Mistral-7B for different unmasking configurations and four SL tasks. We observe a number of interesting phenomena: (1) In most cases, removing the CM in as little as one layer group significantly boosts the F1 score compared to keeping the CM in all layers (configuration 0000); (2) It is rarely the case that removing CM from all layer groups (configuration 1111), compared to removing it only from some layer groups, yields the highest score on evaluation sets; (3) Depending on the task, the boosts from the CM removal can vary (highest for NER, lowest for chunking) and deviate from minimal (NER) to substantial (TC); (4) Mistral-7B shows superior performance over Llama2-7B model across tasks and best configurations; (5) The shapes of validation and test curves show high overlap across configurations for a fixed model and dataset.

Unmasking gains are the highest for NER, and the results are the most consistent for CoNLL03 data (NER and chunking tasks). This aligns perfectly with the values of the RDRR (cf. Section 4), suggesting that labeled NER spans depend more on the right context than on the left (cf. Table 1). We generally observe that unmasking layers closer to the model's output yields higher gains than unmasking layers closer to the model's input, except for ATE+ATP and TC tasks. Such a regularity would align with the finding from the literature that higher layers in neural language models are more localized and task-specific (Wu et al., 2020). To verify whether such a regularity generally holds, we conduct a one-sided two-sample t-test for the difference of F1 score means on validation and test sets per task. We compare the means of each unmasking configuration, which contains at least one unmasked layer group, with all other possible combinations where at least one additional layer group is unmasked closer to the model's output. For example, we compare the F1 score of unmasking configuration 0100 to configurations {0100, 0101, 0110, 0111}. We repeat the same procedure for the following configurations: 0100, 0110, 1000, 1010, 1100, and 1110. The differences are significant for NER and chunking evaluation sets ($p < .01$) but not significant for ATE+ATP and TC evaluation sets ($p > .01$). Applying CM removal to individual layers as opposed to layer groups could give even larger F1 score boosts. The performance of Llama2-7B on the ATE+ATP task displays an unusual pattern where better performance is achieved when CM is preserved in all layer groups, as opposed to its complete removal. Here, allowing the model to "look right" in all layers hurts the performance, which aligns with its low RDRR value. This trend is not present for any other task and model combination. Furthermore, the Rest14 dataset is the only one where higher scores were achieved on the test set than on the validation set. This could be attributed to taking a random sample of training data as a validation set. Micro F1 score deviations across evaluation sets are the largest on ATE+ATP and TC tasks, possibly due to the overall low number of examples per evaluation set for Rest14 and ACE05 datasets (<1000).

5.2 Comparison with Baselines

Table 2 shows the performance scores of the best unmasking configurations and the strong encoder and IT baselines. We compare against RoBERTa PLMs fine-tuned for SL task and instruction-tuned Llama2-7B and Mistral-7B LLMs. The standard 0000 and 1111 configurations are compared with the best configurations per model and task. We also report the Pearson correlation coefficient ρ between all validation and test unmasking configurations. For all SL tasks and the Llama2-7B model, we observe a consistent improvement of best configurations over configuration 1111. Similar holds for Mistral-7B, except for chunking, where 1111 yields the highest F1 scores. Conforming to the findings from Scaria et al. (2023), IT is most beneficial for the ATE+ATP task, outperforming any unmasking configuration on evaluation sets. RoBERTa-large surpasses all other models on the ACE05 validation set but fails to do so on the test set. RoBERTa baselines achieve high scores on SL tasks, except for ATE+ATP. Training with QLoRA combined with layer group unmasking creates high-performing and compact models, requiring a small number of additional parameters trained for each SL task. Correlations between configurations on evaluation sets are high for each task, while models trained on ACE05 exhibit the lowest ρ. A high overall ρ indicates that the optimal unmasking configuration can be determined using the validation set.

5.3 Comparison with SOTA IE Models

Our results are competitive with SOTA. However, a fair comparison is challenging due to differences in training and evaluation. For example, the SOTA F1 score for CoNLL03 NER, reported by Wang et al. (2021), is 0.946. This result, however, was obtained using a model trained on the merged training and validation sets. Strong results on CoNLL03 NER were reported by Liu et al. (2022), reaching an F1 score of 0.941 without task-specific feature engineering, relying solely on a conditional language model with explicit modeling of the target structure. Further, the authors whose work we build upon, Li et al. (2023), report SOTA results on CoNLL03 NER. The score they report is 0.932, which is competitive, but not SOTA. Upon code inspection, we found that they truncate sequences longer than 64 tokens to fit the data into GPU memory. In contrast, we managed to keep the sequence length at 128 tokens. These decisions have a significant impact on overall performance. Although the reported results in related work are SOTA or close to SOTA, these studies are representative of a common problem in the field, namely the fact that the important evaluation details are not always communicated properly (micro vs. macro F1 score, token- vs. span-based evaluation, and evaluating the model prediction on the first token of tokenized words from the input sequence vs. on all tokens). Scaria et al. (2023) achieve SOTA F1 score of 0.928 on Rest14 dataset and ATE task with IT. Yang and Li (2021) report SOTA macro F1 of 0.863 on ATP task with DeBERTa model (He et al., 2020). The SOTA F1 score of 69.8 on ACE05 for TC is achieved by (Wang et al., 2022a).

5.4 Investigating the Effect of Scale

We observed gains upon CM removal on a 7B parameters scale. Our experiments prompt the question of whether similar findings would apply to models of different parameter scales. More specifically, whether CM removal from a small CLM-

Model			CoNLL03 NER		Rest14 ATE+ATP		ACE05 Trigger Clf.		CoNLL03 Chunking	
Llama2-7B-SL			Valid F1	Test F1	Valid F1	Test F1	Valid F1	Test F1	Valid F1	Test F1
			$\rho = 0.999$		$\rho = 0.994$		$\rho = 0.806$		$\rho = 0.998$	
Unmask Config.		0000	0.788	0.744	0.430	0.482	0.591	0.574	0.875	0.864
		1111	0.936	0.908	0.345	0.402	0.635	0.584	0.892	0.880
	Best	0100	–	–	0.488	0.546	–	–	–	–
		0101	–	–	–	–	–	–	–	0.885
		0110	0.947	0.920	–	–	–	–	0.898	–
		0111	–	–	–	–	0.653	–	–	–
		1100	–	–	–	–	–	0.616	–	–
Mistral-7B-SL			Valid F1	Test F1	Valid F1	Test F1	Valid F1	Test F1	Valid F1	Test F1
			$\rho = 0.999$		$\rho = 0.983$		$\rho = 0.899$		$\rho = 0.994$	
Unmask Config.		0000	0.789	0.756	0.484	0.553	0.607	0.584	0.882	0.872
		1111	0.953	**0.927**	0.581	0.641	0.668	0.626	**0.904**	**0.892**
	Best	0110	–	–	0.597	0.659	–	–	–	–
		0111	**0.956**	–	–	–	–	–	–	–
		1101	–	–	–	–	–	**0.639**	–	–
		1110	–	**0.927**	–	–	0.676	–	–	–
RoBERTa-base-SL			0.897	0.883	0.313	0.369	0.609	0.508	0.889	0.877
RoBERTa-large-SL			0.924	0.900	0.403	0.474	**0.698**	0.628	0.891	0.877
Llama2-7B-IT			0.778	0.771	0.523	0.608	0.375	0.347	0.833	0.818
Mistral-7B-IT			0.897	0.887	**0.646**	**0.733**	0.477	0.461	0.873	0.860

Table 2: Validation and test micro F1 SL scores for Llama2-7B and Mistral-7B models with various unmasking configurations (0000, 1111, and other configurations which surpass the F1 score of 1111 over SL datasets – denoted as Best) are in the upper table part. The results for SL encoders and IT baselines are in the lower table part. The best results by dataset and evaluation set are in **bold**. For Llama2-7B and Mistral-7B, we report the Pearson correlation coefficient between validation and test unmasking configurations (ρ). All results are averages over five runs.

based decoder would exhibit the same trend, and also whether MLM-based pre-training is more beneficial for success on SL tasks when the number of parameters, pre-training data and steps, and all other hyperparameters between decoders and encoders are equal. To investigate this, we consider two models of comparable size – a small CLM-based decoder and an MLM-based encoder. We randomly initialize small LMs with four decoder or encoder blocks following RoBERTa-base architecture with a language modeling head on top and a newly initialized embedding matrix with the size of RoBERTa-base's vocabulary. We inherit all other RoBERTa-base hyperparameters and produce a small LM with 68M parameters. We use the RoBERTa-base tokenizer to pre-process the BookCorpus dataset (Zhu et al., 2015) consisting of 74M sentences and use this dataset to pre-train small LMs. We train the small decoder and encoder with CLM and MLM, respectively, with AdamW (Loshchilov and Hutter, 2017) optimizer, a cosine annealing learning rate scheduler, eight gradient accumulation steps, and bfloat16 precision with a batch size of 64 and a learning rate of $2\mathrm{e}{-4}$. We save the model weights immediately after random initialization and then at five equally spaced intervals throughout each epoch (51 checkpoints). The training continues for a fixed number of 10 epochs (roughly 200K steps). After pre-training, we load the weights of each checkpoint, replace the LM head with the SL head for the appropriate task, and fine-tune all parameters for five epochs on a task-specific training set with a batch size of 16. We fine-tune three variants: a pre-trained encoder, a pre-trained decoder with CM during fine-tuning, and a pre-trained decoder without CM. We report averages over five fine-tuning runs, with the last model from each run evaluated on the validation set. More details are provided in Appendix A.2.

The results in Figure 3 reveal that removing the CM on the 68M parameters scale produces no gains. On average, Decoder Unmask performs worse than Decoder Mask. Moreover, encoders struggle to keep up with decoders until around the 20th checkpoint (fourth pre-training epoch), when they start prevailing on all SL tasks except TC. MLM training

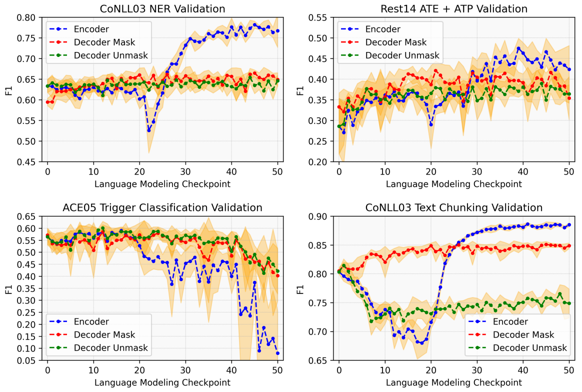


Figure 3: Validation set micro F1 SL scores of small pre-trained MLM-based encoder and CLM-based decoder models after fine-tuning on SL tasks training data starting from particular LM checkpoint. Decoder Unmask: model pre-trained with CLM and a CM but fine-tuned without the CM. All results are averages over five runs. The shaded area represents the standard deviation.

for over four epochs drastically hurts performance on ACE05. The score RoBERTa-base achieves on ACE05 (cf. Table 2) is close to the best small encoder. The gains from continued MLM training on ACE05 become less important than the overall number of model parameters. This can be explained by the fact that RoBERTa-base and RoBERTa-large were trained for 500k steps (Liu et al., 2019), and RoBERTa-large is drastically better on ACE05.

6 Conclusion

The landscape of PLMs shifted from encoder to decoder dominance in various NLU tasks. While large decoder models show remarkable performance even without fine-tuning, challenges persist in IE tasks, typically formulated as sequence labeling. In a series of experiments, we showed that LLMs can yield performance competitive with SOTA on sequence labeling IE tasks when the causal mask is removed from specific decoder blocks. Future work should investigate CM removal for individual layers, analyze task-specific differences in CM removal, and investigate the options for predicting the impact of CM removal without requiring extensive fine-tuning. Promising alternatives to trying out all the unmasking configurations could involve leveraging dynamic programming or estimating a network's final performance without retraining, for example, using methods that rely on the Fisher information, similar to the FIT method by Zandonati et al. (2022). Additionally, performance estimation for CM removal from a layer could be improved using representation similarity analysis methods such as centered kernel alignment (Kornblith et al., 2019). These approaches might reduce the number of unmasking configurations that need to be evaluated to find the best one. Comparisons with encoders stress the significance of architecture and scale in SL tasks. A noticeable gap exists between small-scale encoder and decoder models, where decoder models do not benefit from CM removal at a small scale.

7 Limitations

In our experiments, we ensure reliability by averaging performance scores over five runs with different random seeds. Increasing the sample size for averaging would enhance reliability further. However, maintaining fixed learning rates and other hyperparameters across experiments might have led to suboptimal adaptation for sequence labeling tasks. Exploring additional unmasking configurations could provide valuable insights and potential improvements. While numerous open LLMs are available, we focus solely on Llama2 and Mistral, both with 7B parameters. Enhancing prompt templates used for instruction tuning could boost overall instruction tuning performance. Given more computing resources, experimenting with larger models would be feasible. Additionally, our experiments were limited to English-language datasets. Extending the analysis to sequence labeling tasks in other languages and incorporating more diverse datasets could yield further insights. Finally, the main limitation of our work is the fact that one needs to try out all the possible unmasking configurations to find the best one on the validation set. Trying out all the possible configurations requires extensive fine-tuning.

References

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2Vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*.

Terra Blevins, Hila Gonen, and Luke Zettlemoyer. 2023. Prompting language models for linguistic structure. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6649–6663, Toronto, Canada. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind

Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. *arXiv preprint arXiv:2305.14314*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yifan Ding, Michael Yankoski, and Tim Weninger. 2024. Span-oriented information extraction–A unifying perspective on information extraction. *arXiv preprint arXiv:2403.15453*.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Hao Fei, Shengqiong Wu, Jingye Li, Bobo Li, Fei Li, Libo Qin, Meishan Zhang, Min Zhang, and Tat-Seng Chua. 2022. LasUIE: Unifying information extraction with latent adaptive structure-aware generative language model. *Advances in Neural Information Processing Systems*, 35:15460–15475.

Ridong Han, Tao Peng, Chaohao Yang, Benyou Wang, Lu Liu, and Xiang Wan. 2023. Is information extraction solved by ChatGPT? An analysis of performance, evaluation criteria, robustness and errors. *arXiv preprint arXiv:2305.14450*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: Decoding-enhanced BERT with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength natural language processing in Python.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. NV-Embed: Improved techniques for training LLMs as generalist embedding models. *arXiv preprint arXiv:2405.17428*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. *arXiv preprint arXiv:2109.02846*.

Zongxi Li, Xianming Li, Yuzhang Liu, Haoran Xie, Jing Li, Fu-lee Wang, Qing Li, and Xiaoqin Zhong. 2023. Label supervised LLaMA finetuning. *arXiv preprint arXiv:2310.01208*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Tianyu Liu, Yuchen Eleanor Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. 2022. Autoregressive structured prediction with language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 993–1005, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Ilya Loshchilov and Frank Hutter. 2016. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Jie Lou, Yaojie Lu, Dai Dai, Wei Jia, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2023. Universal information extraction as unified semantic matching. *arXiv preprint arXiv:2301.03282*.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

Maitrey Mehta, Valentina Pyatkin, and Vivek Srikumar. 2024. Promptly predicting structures: The return of inference. *arXiv preprint arXiv:2401.06877*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3470–3487, Dublin, Ireland. Association for Computational Linguistics.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Chaoxu Pang, Yixuan Cao, Qiang Ding, and Ping Luo. 2023. Guideline learning for in-context information extraction. *arXiv preprint arXiv:2310.05066*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. *arXiv preprint arXiv:2101.05779*.

Yang Ping, JunYu Lu, Ruyi Gan, Junjie Wang, Yuxiang Zhang, Pingjian Zhang, and Jiaxing Zhang. 2023. UniEX: An effective and efficient framework for unified information extraction via a span-extractive perspective. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16424–16440, Toronto, Canada. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Kevin Scaria, Himanshu Gupta, Saurabh Arjun Sawant, Swaroop Mishra, and Chitta Baral. 2023. InstructABSA: Instruction learning for aspect based sentiment analysis. *arXiv preprint arXiv:2302.08624*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Erik F. Tjong Kim Sang 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*, pages 142–147.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. 2022a. DeepStruct: Pretraining of language models for structure prediction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 803–823, Dublin, Ireland. Association for Computational Linguistics.

Liwen Wang, Rumei Li, Yang Yan, Yuanmeng Yan, Sirui Wang, Wei Wu, and Weiran Xu. 2022b. InstructionNER: A multi-task instruction-based generative framework for few-shot ner. *arXiv preprint arXiv:2203.03903*.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023a. GPT-NER: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, et al. 2023b. InstructUIE: Multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Automated concatenation of embeddings for structured prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2643–2660, Online. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022c. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

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

John Wu, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2020. Similarity analysis of contextual word representation models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4638–4655, Online. Association for Computational Linguistics.

Heng Yang and Ke Li. 2021. Improving implicit sentiment learning via local sentiment aggregation. *arXiv preprint arXiv:2110.08604*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Ben Zandonati, Adrian Alan Pol, Maurizio Pierini, Olya Sirkin, and Tal Kopetz. 2022. Fit: A metric for model sensitivity. *arXiv preprint arXiv:2210.08502*.

Rongzhi Zhang, Yue Yu, and Chao Zhang. 2020. SeqMix: Augmenting active sequence labeling via sequence mixup. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8566–8579, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Wenxuan Zhou and Muhao Chen. 2021. Learning from noisy labels for entity-centric information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5381–5392, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tong Zhu, Junfei Ren, Zijian Yu, Mengsong Wu, Guoliang Zhang, Xiaoye Qu, Wenliang Chen, Zhefeng Wang, Baoxing Huai, and Min Zhang. 2023. Mirror: A universal framework for various information extraction tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8861–8876, Singapore. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Appendix

A.1 Experimental Setup Details

Since RoBERTa requires tokenization with added prefix space, we enforce the same for open LLMs' tokenizers. We use the end-of-sequence token for the models with no pre-set padding token. The cross-entropy loss is adjusted to consider only the first token of each tokenized word from the input sequence. We report all results as averages of five runs and use $seeds = \{120, 121, 122, 123, 124\}$.

For IT experiments, we generate the outputs for evaluation with default generation settings for the Llama2-7B model. To speed up the generation, we decrease the total maximum length of the input instruction prompt combined with newly generated tokens to 1024. The same generation config is used for Mistral-7B.

A.2 Training Small Language Models Details

We use a RoBERTa-base tokenizer with added prefix space. Tokenized BookCorpus sentences are grouped to form chunks of size 512 for either MLM-based pre-training of small encoder or CLM-based pre-training of small decoder LM. MLM probability is set at 0.15. Models are trained with AdamW. Its parameters are fixed to $\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 1\text{e}{-5}, \lambda = 0.1$ and we apply gradient clipping to 1.0. We pre-train both models with the same seed. We fine-tune for SL tasks five times and use $seeds = \{120, 121, 122, 123, 124\}$.

A.3 Instruction Tuning Details

Task (dataset)	Instruction example for training	Instruction example for evaluation
NER (CoNLL03)	### Instruction: please extract named entities and their type from the input sentence, all entity types are in options ### Options: person, location, organization, miscellaneous ### Sentence: " What we have to be extremely careful of is how other countries are going to take Germany 's lead , " Welsh National Farmers ' Union (NFU) chairman John Lloyd Jones said on BBC radio . ### Response: Germany:location;Welsh National Farmers ' Union:organization;NFU:organization;John Lloyd Jones:person;BBC radio:organization	### Instruction: please extract named entities and their type from the input sentence, all entity types are in options ### Options: person, location, organization, miscellaneous ### Sentence: " What we have to be extremely careful of is how other countries are going to take Germany 's lead , " Welsh National Farmers ' Union (NFU) chairman John Lloyd Jones said on BBC radio . ### Response:
ATE+ATP (Rest14)	### Instruction: please extract aspect terms and their polarity from the input sentence, all polarity types are in options ### Options: positive, negative, neutral, conflict ### Sentence: The lobster sandwich is $ 24 and although it was good it was not nearly enough to warrant that price . ### Response: lobster sandwich:conflict;price:negative	### Instruction: please extract aspect terms and their polarity from the input sentence, all polarity types are in options ### Options: positive, negative, neutral, conflict ### Sentence: The lobster sandwich is $ 24 and although it was good it was not nearly enough to warrant that price . ### Response:
Trigger Classification (ACE05)	### Instruction: please extract events and their types from the input sentence, all event types are in options ### Options: merge organization, start organization, declare bankruptcy, end organization, grant pardon, extradite, execute, impose fine, conduct trial hearing, issue sentence, file appeal, convict, file lawsuit, release on parole, arrest and send to jail, charge and indict, acquit, participate in protest or demonstration, attack, contact via written or telephone communication, meet, start position, elect, end position, nominate, transfer ownership, transfer money, marry, divorce, be born, die, sustain injury, transport ### Sentence: In his previous letter home , Apache pilot Joe Bruhl did n't tell his family the full details about his first combat mission . ### Response: tell:contact via written or telephone communication;combat:attack	### Instruction: please extract events and their types from the input sentence, all event types are in options ### Options: merge organization, start organization, declare bankruptcy, end organization, grant pardon, extradite, execute, impose fine, conduct trial hearing, issue sentence, file appeal, convict, file lawsuit, release on parole, arrest and send to jail, charge and indict, acquit, participate in protest or demonstration, attack, contact via written or telephone communication, meet, start position, elect, end position, nominate, transfer ownership, transfer money, marry, divorce, be born, die, sustain injury, transport ### Sentence: In his previous letter home , Apache pilot Joe Bruhl did n't tell his family the full details about his first combat mission . ### Response:
Text Chunking (CoNLL03)	### Instruction: please extract chunks and their type from the input sentence, all chunk types are in options ### Options: noun phrase, verb phrase, prepositional phrase, adverb phrase, subordinated clause, adjective phrase, particles, conjunction phrase, interjection, list marker, unlike coordinated phrase ### Sentence: Rare Hendrix song draft sells for almost $ 17,000 . ### Response: Rare Hendrix song draft:noun phrase;sells:verb phrase;for:prepositional phrase;almost $ 17,000:noun phrase	### Instruction: please extract chunks and their type from the input sentence, all chunk types are in options ### Options: noun phrase, verb phrase, prepositional phrase, adverb phrase, subordinated clause, adjective phrase, particles, conjunction phrase, interjection, list marker, unlike coordinated phrase ### Sentence: Rare Hendrix song draft sells for almost $ 17,000 . ### Response:

Table 3: Instruction tuning examples from the training sets of the four SL datasets.