Abstract

Structured Natural Language Processing (XNLP) is an important subset of NLP that entails understanding the underlying semantic or syntactic structure of texts, which serves as a foundational component for many downstream applications. Despite certain recent efforts to explore universal solutions for specific categories of XNLP tasks, a comprehensive and effective approach for unifying all XNLP tasks long remains underdeveloped. Meanwhile, while XNLP demonstration systems are vital for researchers exploring various XNLP tasks, existing platforms can be limited to, e.g., supporting few XNLP tasks, lacking interactivity and universality. To this end, we propose an advanced XNLP demonstration system, where we leverage LLM to achieve universal XNLP, with one model for all with high generalizability. Overall, our system advances in multiple aspects, including universal XNLP modeling, high performance, interpretability, scalability, and interactivity, offering a unified platform for exploring diverse XNLP tasks in the community.

1 Introduction

XNLP has been referred to as a special form of NLP tasks that involves holistically analyzing and interpreting the underlying semantic or syntactic structure within a text, such as Syntactic Dependency Parsing (Nivre, 2003), Information Extraction (Wang and Cohen, 2015), Coreference Resolution (Lee et al., 2017), and Opinion Extraction (Pontiki et al., 2016), etc. Figure 1 (upper part) illustrates some representative XNLP tasks under different categories. XNLP has been infrastructural for a wide range of downstream NLP applications, such as Knowledge Graph Construction (Bosselut et al., 2019), Empathetic Dialogue (Rashkin et al., 2019), and more newly-emerging applications and techniques (Tang et al., 2020).

As the key common characteristics, all the XNLP tasks have revolved around predicting two key elements from input: 1) textual spans and 2) relations between spans (Fei et al., 2022b), as depicted in the Figure 1 (lower part). Traditional efforts for XNLP have treated each task independently, which has led to limited utilization of shared features among XNLP tasks (He et al., 2019), and sub-optimal model generalization across different datasets (Chauhan et al., 2020), such as cross-language and cross-domain scenarios. In this paper, we emphasize the importance of Model Unification as a crucial topic in NLP. By unifying various NLP tasks under the XNLP framework, we can take advantage of the shared characteristics among tasks, leading to better model generalization and improved performance in realistic scenarios of product deployment.

Despite recent certain efforts in exploring universal solutions for some categories of XNLP tasks, such as Unified Sentiment Analysis (Chen and...

XNLP is online: https://xnlp.haofei.vip

Video demonstration at https://youtu.be/bOc-9HELEVw
Qian, 2020; Fei et al., 2022a), and Universal Information Extraction (UIE; Lu et al., 2022; Fei et al., 2022b), a comprehensive and effective approach for unifying all XNLP tasks has not been fully established. Fortunately, Large Language Models (LLMs) (Vaswani et al., 2017; Raffel et al., 2020) present a potential solution for unification across all XNLP tasks. There is a recent development in the form of LLMs, e.g., ChatGPT (Ouyang et al., 2022), LLaMA (Touvron et al., 2023) and Vicuna (Peng et al., 2023), that have shown promising advancements in NLP and other fields. LLMs, with sufficient sizes of model and data, have demonstrated impressive generalization capabilities, well supporting the idea of “One model for all” (OpenAI, 2023). In this work, we propose taking advantage of LLMs to achieve universal XNLP, addressing the lack of a well-defined and holistic approach.

On the other hand, demonstration systems play a crucial role for researchers (especially beginners) exploring various XNLP tasks, providing a platform to analyze and understand the functionalities of different NLP components and their applications. While there are existing widely-used XNLP demo systems, such as CoreNLP2, AllenNLP3, we have observed several key issues with them: 1) limited to only a few specific tasks; 2) lacking interactive and extensible features, making it challenging to support dynamic growth in new XNLP tasks; 3) not universal systems, requiring separate models for each task, which can lead to increased overhead. To address these limitations, this work aims to build an advanced platform that provides superior XNLP demonstrations and benefits the broader NLP community.

In summary, our system advances in the following aspects.

1) Universalness
   • Our XNLP system takes the existing open-source LLMs as the backbone engine with excellent generalization capabilities, enabling unified prediction of various XNLP tasks, leading to a streamlined and cohesive XNLP ecosystem.
   • The LLM-based system supports end-to-end predictions for complex structured tasks, regardless of whether the spans are nested or discontinuous, making it versatile and adaptable to different linguistic structures.

2) High Performance
   • Our system is capable of few-shot or weakly-supervised learning. Having undergone extensive pre-training, LLMs do not require in-domain fine-tuning on specific task data.
   • Our system supports open-label and vocabulary predictions, utilizing LLM’s generalization capabilities to discover new labels and vocabs with superior out-of-domain generalization.
   • Our approach naturally lends itself to cross-lingual, code-switching, and cross-domain settings.

3) Scalability & Interpretability & Interactivity
   • The system allows dynamic addition and definition of new tasks, requiring users only to provide demonstrations for the new tasks.
   • Predictions generated by our system are interpretable, as LLMs are able to provide rationales for their decisions, explaining why a specific result is produced.
   • The system enables user-machine interaction, empowering users to provide feedback, thereby allowing the system to refine its predictions based on user input.

2 Related Work

2.1 Structured NLP

Over the last few decades, XNLP has garnered significant research attention, with several works addressing specific aspects of XNLP tasks, spanning from linguistic/syntactic parsing (Kitaev and Klein, 2018), to information extraction (Mikheev et al., 1999), to semantic analysis (He et al., 2017), and to sentiment analysis & opinion mining (Wu et al., 2021). Prior studies and efforts have been paid and achieved notable developments for each of the XNLP tasks, such as Syntactic Dependency Parsing (Nivre, 2003), Information Extraction (Wang and Cohen, 2015), Coreference Resolution (Lee et al., 2017), and Opinion Extraction (Pontiki et al., 2016), etc. Different XNLP tasks may have different specific task definitions, while prediction formats of all the XNLP tasks can be reduced to the same prototype: the term extraction and relation detection (Lu et al., 2022; Fei et al., 2022b).

Demonstration for XNLP. The development of demonstration platforms has been crucial for educational and academic purposes, e.g., aiding researchers to explore various tasks and gaining hands-on experiences. Existing widely-employed
open demo systems for XNLP include CoreNLP\(^4\), AllenNLP\(^5\) and Explosion.ai\(^6\) etc. While offering user-friendly web interface for users to access a set of XNLP functionalities, there remain certain limitations, such as lacking flexibility for incorporating new tasks, non-universality for model and cross-domain generalization.

### 2.2 Model Unification

There have been notable efforts to explore universal modeling for a type of NLP tasks (Chen and Qian, 2020; Fei et al., 2022a; Lu et al., 2022; Fei et al., 2022b), showcasing the benefit and potential of model unification, e.g., better leverage of shared characteristics and knowledge across tasks, simplified model maintenance, and enhanced system efficiency. However, a comprehensive and effective approach for unifying all XNLP tasks remains under-investigated. In this work, by capitalizing on LLM’s robustness and broad applicability, we aim to pave the way for an advanced unified framework capable of handling diverse XNLP tasks effectively.

### 3 System Design

#### Architecture overview.

We design our XNLP demo system into a web interface form. Built based on the Django\(^7\) framework, XNLP divides the functions into the **frontend** module and the **backend** module. As shown in Figure 2, the frontend takes user inputs and displays the visualization of outputs, and the backend provides task prediction services with LLM as its core engine, based on the in-context learning paradigm. Also, it is possible for multi-turn interactions between frontend and backend.

#### 3.1 Backend

**Backbone LLM.** Among a list of open-source LLMs, we consider the Vicuna-13B\(^8\) as our backbone. Trained by fine-tuning LLaMA (Touvron et al., 2023) on user-shared conversations collected from ShareGPT, Vicuna has achieved more than 90% of OpenAI ChatGPT’s (Peng et al., 2023) quality in user preference tests.

**In-context learning.** To elicit LLM to induce task predictions, we build in-context prompts. We note that to ensure the support of universal XNLP for any potential tasks and inputs, the prompt template should cover rich and informative information from the user end. Thus, we design the prompt by mainly covering the **task name**, **task description**, **task demonstration**, **task label set**, **executing format**, **input text**, **language** and **domain**.

```plaintext
{Task-desc}
For example, {Task-demo}
Note the task output format should be with this: {Exe-format}
And generally, the desired predicted labels should be within the following given label set: {Task-label}
Now, given a new test input: "{Input-text}”, please do the task of {Task-name}.
Note the input is with {Language} language, and the text is from the {Domain} domain.
Please predict all possible results strictly following the exact given format, without any other output of explanations.
```

Fed with the above prompt, the LLM is ex-
pected to output prediction in the provided format (executing format), with which, the post-process program further parses and polishes the structure result, and packs data to return to the frontend.

**Broad-cover structure-aware instruction tuning.** While LLM’s outputs are sequential, XNLP tasks are highly structured. Thus, we expect the LLM to generate strictly structural results conditioned on sequence inputs. We consider further tuning the LLM with a broad-cover structure-aware instruction tuning mechanism. Instruction tuning is an emergent paradigm of LLM fine-tuning wherein natural language instructions are leveraged with LLM to induce the desired result more accurately. We write the XNLP predictions (outputs) for any input prompt by formatting the predictions into task-agnostic (i.e., task broad-covering) well-formed structure representations as in Fei et al. (2022b).

Structure formatting. As aforementioned, all the XNLP can be unified by predicting two key elements: the term extraction (with the span attribute) and relation detection (with the relation type), as illustrated in Figure 1. To unify all XNLP tasks, we follow Fei et al. (2022b) and design a structure formatter, where all the XNLP task outputs share the same structural representations. As shown in Figure 4, under the structure formatted, all XNLP tasks have been divided into the span extraction, pair extraction and hyper-pair extraction.

3.2 Frontend

As illustrated in Figure 2 (upper part), the frontend of XNLP receives inputs of 1) texts or user feedback or 2) task metadata (pre-defined or user-defined), and exhibits outputs from LLM. Following we mainly describe the key features of the frontend module as listed below.

**Pre-defined XNLP tasks.** To facilitate the user operation, we pre-defined total 22 XNLP tasks, covering four frequent categories, including Syntax Parsing, Information Extraction, Semantic Analysis and Sentiment/Opinion Mining.

**New task definition.** As there are rapidly-emergent XNLP tasks in the NLP community, it is impossible to cover it all in the pre-definition. We thus allow users to define their own XNLP tasks. This can be easily accomplished in our system without much effort, as the LLM has exceptional zero-shot performance and understanding ability. We require from the user only the task name, task description, task demonstration, task label set, executing format.

**XNLP structure visualization.** The key role of XNLP system is the visualization of the task output structure. We employ the open-source brat system\(^9\) to realize this. brat has been shown very popular and effective in rendering structured data, with pretty visualization and stable functions.

**Rationale for explainable task prediction.** Besides the visualization of direct task results, we also display the rationale for each prediction, allowing seeing what and knowing why. This is especially meaningful for the beginners of the researchers for XNLP tasks. To enable this, we just ask LLM “How and why do you make your decision?” after each task prediction.

**Enhancing prediction with user interaction.** To take full advantage of the LLM, we further allow users to interact with our system by providing any feedbacks, so that users can revise the task predictions whenever they feel the results are not incorrect or coincident with their minds. To reach this, we also add another round of query to LLM, by asking “The above prediction is not all right, because Feedback. Please do the task again by carefully taking the feedback here”.

4 System Walkthrough

Figure 3 gives a comprehensive walkthrough of how the system can be operated by users.

- **Step-1.** users select or define a task;
- **Step-2.** users go through (for pre-defined) or fill in (for user-defined) the task prompt;
- **Step-3.** users key in the text to analyze;
- **Step-4.** users submit the text & metadata and request result;
- **Step-5.** users can browse the visualization of task output;
- **Step-6.** users observe the rationale of this result;
- **Step-7.** users can further provide feedback for the system to re-generate result;

Following we demonstrate XNLP system by walking readers through several important functions.

4.1 User-allowed Operations

**Pre-defined XNLP task selection.** For the first step, users should select an XNLP task template

\(^9\)https://brat.nlplab.org/
Figure 3: Screenshot of the XNLP web application, where key functions are annotated.

Figure 4: Structure formatter for universal XNLP, from the 22 system pre-defined pools. The operation is shown in Figure 7 in Appendix §A.

New task definition. Or, user can define their own tasks. As shown in Figure 8 in Appendix §B, users should decide the task name, and fill in the task metadata (task prompt) as shown in Figure 3, and also select a pre-defined task with which the new task shares most similarity.

Language and domain notifications. To enable more accurate predictions, it is better to explicitly notify LLM what language and domain the input has. Figure 10 in Appendix §D and Figure 11 in Appendix §E illustrate the operations, respectively.

Improving/Revising Prediction with User Feedback. Figure 9 in Appendix §C showcase the operation for the multi-turn user interaction.

4.2 Task Visualization

Here we showcase the XNLP task visualizations of real examples via our system. Figure 5 renders the outputs for the four task clusters, with each
showing three representative task results, such as:

**Syntax parsing**, including Part-of-Speech (POS) Tagging, Dependency Parsing and Constituency Parsing.

**Semantic analysis**, including Semantic Role Labeling (SRL), Coreference Resolution, and Intent Recognition and Slot Filling.

**Information extraction**, including Named Entity Recognition (NER), Relation Extraction, and Event Extraction.

**Sentiment/opinion mining**, including Aspect-based Sentiment Analysis (ABSA), Sentiment Triplet Extraction and Opinion Role Labeling.

We can observe from the visualizations that, 1) the structure visualizations are pretty, owing to the use of the brat system; 2) the results of tasks are correct, for which we give the credit to the integration of LLM, and also the broad-cover structure-aware instruction tuning mechanism.

### 5 Performance Evaluation

To quantitatively verify the performance of the backbone LLM on XNLP tasks, we now perform evaluations. We compare the Vicuna (13B) with the ChatGPT over 100 randomly selected test instances of 6 XNLP tasks. The experiments are based on one-shot in-context learning, i.e., with one demonstration as input. Figure 6 shows the comparisons. We see Vicuna has a slightly lower performance than ChatGPT, while Vicuna after broad-cover structure-aware instruction tuning (Vicuna+StruIT) shows results even much better than ChatGPT, with smaller model size (13B vs. 175B).

### 6 Conclusion

We present XNLP, an advanced online demonstration system for interaction and visualization of XNLP tasks. XNLP, built upon LLM, effectively models all the XNLP tasks universally, achieving one model for all in zero-shot or weak supervision. XNLP not only renders the output structures with
delicate visualizations, but also provides rationales for interpretable predictions. Also, XNLP allows the users to define newly emergent XNLP tasks; and enables users to dynamically revise the output with multi-turn interactions. Our XNLP contributes to the community by paving the way for a unified, scalable, and interactive demonstration platform.

Limitations

The focus of this paper was introducing an open online web application (demonstration system) to make the interaction of XNLP tasks available to as many practitioners as possible, but there are a couple of limitations in the system and the model we proposed. First, our system is based on the web service form, with the LLM running at the backend deployed at the online server, where sometimes when the Internet traffic is bad, the user may wait for too long to get the response. Second, as the LLM essentially generates sequential texts of any inputs, there are chances that the output texts include problematic structured formatter (i.e., structural representations, cf. Figure 4). With ill-formed structural representations, it is problematic to parse them into correct data used for rendering into brat visualization, i.e., causing failure prediction. Third, as one of the nature characteristics, LLM may sometimes generate false output, or do not obey the input instructions, which has been called the Hallucination phenomenon (Varshney et al., 2023). In such case, the user experience will be affected. Lastly, the current version of the system is still at a basic stage, and there are functionalities at the user interface level that need further polishing and improvement in subsequent updates.

Ethics Statement

Our XNLP system uses the LLM as backbone. While the Vacuna model is fine-tuned on the pre-trained LLaMA model, which is known to contain some toxic contents (Schick et al., 2021), an internal check does not reveal any toxic generation. However, there is a potential risk that the Vacuna could generate toxic text for users due to the underlying black-box LLM.

References


A Selection of Pre-defined XNLP Tasks
See Figure 7.

B New XNLP Task Definition
See Figure 8.

C Multi-turn Interactions with User Feedback
See Figure 9.

D Text in Different Language
See Figure 10.

E Text in Different Domain
See Figure 11.
Figure 8: Screenshot of defining new XNLP task by the user.
Figure 9: System output before user feedback and after user feedback.
Figure 10: Input text in different languages (Chinese).
Figure 11: Input text in different domains (financial domain).