Masterarbeit
zur Erlangung des akademischen Grades
Master of Arts
der Philosophischen Fakultät der Universität Zürich

Cross-lingual Projection of Text Zoning Labels for Job Advertisements

(with Minor Revisions for Publication)

Verfasser: Wenyuan Wu
Matrikel-Nr: 18-746-867

Referent: Dr. Simon Clematide

Betreuerin: Ann-Sophie Gnehm, Chantal Amrhein

Institut für Computerlinguistik

Abgabedatum: 01.12.2022

Abstract

The study of text zoning for job advertisements from the Swiss Job Market Monitor (SJMM) aims to partially substitute manual annotation by automatic data processing with supervised machine learning to lower data collection costs and extend the research span. Previous work has built and evaluated approaches based on sequence-labeling machine learning approaches like BiLSTM-CRF and BERT language models with success based on the data collected and labeled by SJMM during the past decades. However, the large majority of the training data from SJMM is only available in German, and much less in French, English, and especially Italian, which leads to the labeled data acquisition bottleneck. As a result, the performance of machine learning approaches on the text zoning tasks in non-German languages is relatively reduced. Hence, it is necessary to expand the scope of research to multilingual scenarios, which reflects the actual language use of job advertisement in Switzerland of Switzerland.

The goal of this thesis is to address this problem by testing several approaches for the cross-lingual projection of text zoning labels from job advertisements in German to other languages. The implementation of these approaches is realized by automatically translated and labeled job advertisements, this results in a "Silver Standard" dataset that has comparable training and test splits across languages. Creating silver standard data leverages the injection of original data with XML tags, as well as the API from the DeepL translator. Based on labeled data in non-German languages, the straightforward approach is to project the text zoning labels with the help of statistical and neural word aligners, while the other is to train multi-lingual sequence labeling machine learning models in the same way as the previous work, which is the training process based on multilingual BERT, RoBERTa, domain-adapted variants. The segmentation differs for the training data: sentence-level zone-tagging with and without context, and job-ad-level zone-tagging. Evaluating results on the silver standard data show that approaches involving word aligners have a strong performance, and the neural word aligners improve the label projection accuracy compared to statistical word aligners. It has been observed that sequence labeling models trained on silver standard data produce results that are competitive, with only slight variations in performance. Experiments yielded an average accuracy score of 91% or greater, demonstrating the efficacy and utility of the proposed methods, while providing insight into alleviating the labeled data acquisition bottleneck.

Zusammenfassung

Die Studie zur Text Zoning von Stellenanzeigen aus dem Swiss Job Market Monitor (SJMM) zielt darauf ab, die manuelle Annotation teilweise durch eine automatische Datenverarbeitung mit überwachtem maschinellem Lernen zu ersetzen, um die Kosten für die Datenerfassung zu senken und die Forschungsspanne zu erweitern. In früheren Arbeiten wurden auf der Grundlage der vom SJMM in den letzten Jahrzehnten gesammelten und etikettierten Daten erfolgreich Ansätze des maschinellen Lernens auf der Basis von Sequenz-Labeling wie BiLSTM-CRF und BERT-Sprachmodelle entwickelt und evaluiert. Der Großteil der Trainingsdaten von SJMM ist jedoch nur in deutscher Sprache verfügbar und viel weniger in Französisch, Englisch und vor allem Italienisch, was zu einem Engpass bei der Beschriftung von Daten führt. Infolgedessen ist die Leistung von Ansätzen des maschinellen Lernens bei Text-Zoning-Aufgaben in nicht-deutschen Sprachen relativ gering. Daher ist es notwendig, den Forschungsbereich auf mehrsprachige Szenarien auszuweiten, die den tatsächlichen Sprachgebrauch von Stellenanzeigen in der Schweiz widerspiegeln.

Das Ziel dieser Arbeit ist es, dieses Problem zu adressieren, indem verschiedene Ansätze für die sprachübergreifende Projektion von Text-Zoning-Etiketten aus Stellenanzeigen in Deutsch auf andere Sprachen getestet werden. Die Implementierung dieser Ansätze wird durch automatisch übersetzte und etikettierte Stellenanzeigen realisiert, was zu einem "Silberstandard"-Datensatz führt, der vergleichbare Trainings- und Test-Splits über alle Sprachen hinweg aufweist. Die Erstellung von Silver Standard Daten nutzt die Injektion von Originaldaten mit XML-Tags, sowie die API des DeepL Übersetzers. Auf der Grundlage von beschrifteten Daten in nicht-deutschen Sprachen besteht der einfache Ansatz darin, die Text-Zonenbeschriftungen mit Hilfe von statistischen und neuronalen Wort-Alignern zu projizieren, während der andere Ansatz darin besteht, mehrsprachige Sequenz-Labeling Modelle für maschinelles Lernen auf die gleiche Weise zu trainieren wie die vorherige Arbeit, d.h. der Trainingsprozess basiert auf mehrsprachigen BERT-, RoBERTa- und domänenangepassten Varianten. Die Segmentierung unterscheidet sich für die Trainingsdaten: Zonentagging auf Satzebene mit und ohne Kontext und Zonentagging auf Stellenanzeigenebene. Die Auswertungsergebnisse auf den Silberstandarddaten zeigen, dass Ansätze, die Wortaligner einbeziehen, eine starke Leistung haben, und die neuronalen Wortaligner verbessern die Genauigkeit der Etikettenprojektion im Vergleich zu statistischen Wortalignern. Es wurde festgestellt, dass SSequenz-Labeling Modelle, die auf Silberstandarddaten trainiert wurden, konkurrenzfähige Ergebnisse mit nur geringen Leistungsschwankungen liefern. Die Experimente erbrachten eine durchschnittliche Genauigkeit von 91% oder mehr, was die Wirksam-

keit und den Nutzen der vorgeschlagenen Methoden belegt und gleichzeitig einen Einblick in die Beseitigung des Engpasses bei der Beschriftungsdatenerfassung bietet.

Acknowledgement

I would like to thank my supervisors, Simon Clematide and Ann-Sophie Gnehm, for their ongoing support throughout this thesis. Furthermore, Chantal Amrhein provided help in the early stage of this project. They showed appreciation for my hard work and created an atmosphere of collaboration and enthusiasm. This can then inspire my creativity and help to create a thesis that is interesting and engaging. And many thanks to Milin Zhang for proofreading the present text. Finally, thanks for the company of my cat, Rachael, when the rest of the world is collapsing.

Affiliation The Swiss Job Market Monitor is affiliated with the Institute of Sociology at the University of Zurich. The research team (Dr. Helen Buchs, Yanik Kipfer (MA), Eva Bühlmann (MSc), lic. phil. Ann-Sophie Gnehm, Jan Müller (MA), Felix Busch (PhD)) is headed by Prof. Dr. Marlis Buchmann.

Contents

List of Figures

List of Tables

List of Acronyms

ACL	Association for Computational Linguistics
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-term Memory
BiLSTM-CRF	Bidirectional Long Short-term Memory - Conditional Random Field
MT	Machine Translation
MNT	Neural Machine Translation
NER	Named Entity Recognition
NLP	Natural Language Processing
POS	Part-Of-Speech
SJMM	Swiss Job Market Monitor
SOTA	State-of-the-art
XML	eXtensible Markup Language

1 Introduction

The objective of this master's thesis is to address the issue of labeled data acquisition limitation from the viewpoint of cross-lingual annotation projection. The data for the case study was obtained from the Swiss Job Market Monitor, and consists of job advertisements with text zoning segmentation annotations in the German language. Experimental methods employed include neural machine translation, word alignment, multi-lingual language models, and sequence labeling machine learning models. Experiments are conducted by producing silver standard data derived from the original gold standard data and assessing various methods for transferring text zoning labels between different language pairs. This thesis begins by providing an introduction to the pertinent background information, as well as outlining the motivation, research questions, and structure of the thesis.

1.1 Motivation

Switzerland is a multilingual country, and languages beyond official ones are of the same great importance for academic research. Unfortunately, the text zoning corpus collected and annotated by SJMM is mainly available in German and much less in French, English, and especially Italian, which brings various restrictions in extending the research span and leads to a reduced performance of text zoning in non-German languages. The performance of a machine learning system previously depends heavily on a large amount of labeled training data. Newer machine-learning techniques have proposed advanced approaches. For instance, extremely large language models can perform competitively on downstream tasks with far less task-specific data than would be required by smaller models (Brown et al., 2020), and zero-shot transfer learning in modern NLP shows promising results in classification tasks as well (Weber and Steedman, 2021).

However, language models in large sizes remain impractical for real-world scenarios due to limited GPU memory, and they are also not available in the same quality as for English. Besides, zero-shot transfer learning is less researched for cross-lingual settings. This project instead tries to address the challenge of cross-lingual annotation projection by projecting labels between languages via word aligners and automatically generating labeling data via trained multi-lingual zone taggers, both base on automatically created silver standard data. The experiment results could shed light on further research of text zone labeling and benefit the text mining work of SJMM. Additionally, the specific case study about cross-lingual text zoning label projection could be generalized to other information extraction or text mining tasks in NLP in multilingual settings and hopefully contributes to solving the problem of labeled data acquisition bottleneck.

1.2 Cross-lingual Annotation Projection

Cross-lingual transfer is the underlying concept of annotation projection across languages. It can be argued that, by utilizing alignment or other techniques, annotations sourced from a text in one language can be projected onto a corresponding text in another language, thereby creating a newly annotated corpus for the latter language. In recent years, due to advances in machine translation quality and range of applications, cross-lingual annotation projection has been garnering increasing academic interest. Translation systems are restricted to accepting only plain text as input. Still, the annotation should be correctly projected because either markups or annotations exist by default in some machine translation tasks like the webpage translation in HTML format or computer-assisted translation, where much extra information is annotated alongside the text. Projection of named entity recognition (NER) annotations across languages is widely researched (Ehrmann et al., 2011; Weber and Steedman, 2021; Sluyter-Gäthje et al., 2020). In addition, there have been proposed strategies for the transfer of markup in the field of translation and localization services, involving word alignment and projection algorithms (Galassi et al., 2020; Zenkel et al., 2021). An abundance of research has been conducted to explore the efficacy of cross-lingual datasets translated from English to solve tasks that have English datasets in other languages (Conneau et al., 2018).

The generation of silver standard data is attained by the reformatting of the source data with XML tags, as well as the utilization of the Application Programming Interface (API) from a machine translation engine. The Extensible Markup Language (XML) is a text-based language utilized for a variety of purposes in computational linguistics and other scientific fields (Bray et al., 2008). XML tags are instrumental in delineating the boundaries of an element within an XML document, which serves as the basis for XML. More specifically, XML tags identify the data and are used to store and organize the data, which can be utilized to exchange information between systems, i.e., in this case, between languages. The suitability of XML tags for encoding the text span annotation information into plain text and transferring it to a machine translation engine is a key factor for the successful completion of this thesis project. The XML tag injection pipeline facilitates the accurate and effortless recovery of text span annotations from translation output, regardless of the language pairs or translation engine used.

1.3 Machine Translation

Since its initial conception in the 1960s, machine translation has evolved considerably, with current state-of-the-art performances being achieved through neural machine translation (NMT) approaches (Tan et al., 2020). The utilization of transformer architectures and attention mechanisms, in combination with the availability of large-scale training corpora, has enabled the development of machine translation models for specific language pairs that are suitable for both academic and industrial applications. Consequently, this project takes advantage of the NMT-based DeepL translation engine[1], which is sufficiently effective to overlook the remaining translation issues to a certain degree and concentrate on exploring the transmission of cross-language annotations. DeepL provides an API that can be easily utilized in Python and offers industry-leading machine translation capabilities. Of particular note, it is able to process XML tags in both the input and output text, thereby facilitating the creation of silver standard data.

1.4 Word Alignment

Word alignment is the task of finding the correspondence between source and target words in a pair of sentences that are translations of each other. Word alignment has been shown to be a beneficial outcome of advancements in machine translation, often serving as an ancillary product of the machine translation process. Moreover, word alignment plays an essential role in translation quality and fine-grained NLP tasks downstream. This study incorporates two popular statistical and neural word alignment algorithms and evaluates their efficacy in terms of text zone label projection. This methodology is denoted as the align and project technique, and Chapter 4.1 provides a summary of the implementation along with associated specifics.

[1] `https://www.deepl.com/en/docs-api`

1.5 Text Zoning

Text zoning is the process of dividing a job advertisement into distinct sections, each of which can be characterized by a different content domain, such as a description of the company, the reason for the vacancy, and the job agency description (Gnehm, 2018). The tokens within the job advertisement text will be classified into one of the eight predefined zones, which is essentially a task of annotating text spans. The job advertisement text and the corresponding annotation originate from the continuous work of the Swiss Job Market Monitor, which is the research team from the University of Zurich. The research team is investigating the use of text mining to analyze labor market demand and to generate practical insights from job adverts. In Section 3.1, a comprehensive explanation of the corpus of job advertisements and text zoning labels is presented.

A vital processing step in the information extraction pipelines is the token-based segmentation of the text of job ads into domain-specific text zones. Previous research has made great progress in implementing machine learning approaches for automatic text zone labeling tasks (Gnehm, 2018; Gnehm and Clematide, 2020; Gnehm et al., 2022). However, due to the small amount of training data in languages other than German, the trained machine learning models had less satisfactory performance when labeling data in non-German languages. This thesis project concentrates on testing several approaches for the cross-lingual projection of text zoning labels from job advertisements in German to other languages, which are English, French, and Italian. The implementation of these approaches is accomplished by machine-translated and -labeled job advertisements, which is *de facto* the creation of a multilingual “silver standard dataset”.

This project leverages the automatically generated silver standard data to train multilingual sequence labeling models (zone tagger) and evaluates the models' performance on text zone label accuracy. Besides, the *align-and-project* approach refers to the cross-lingual annotation projection via word alignment. Both are further elaborated on in Chapter 4. As for the experiments on machine learning, different structures, type of training materials, and strategies were also implemented to grab a wider view in terms of model performance, including the basic model training with pre-trained word embeddings in the unit of whole job advertisements or sentences, fine-tuning transformer language models and 2-phase training for potential improvements. The trained multilingual zone taggers show comparable results to the previous work in terms of the accuracy of text zone labels on German data as well as on non-German data, which proves the legibility of the approaches applied in this project.

1.6 Research Questions

The goal of this work is to test approaches for cross-lingual projection of text zoning labels from German job advertisements to other languages. Based on the methods, the research questions are related to four topics:

1. the word alignment approach
2. the zone tagger approach
3. the different types of training data for the zone tagger
4. the benefits brought by the alternation of training techniques

Moreover, this work attempts to analyze the errors in text zoning in terms of languages or, more specifically, the gold standard data versus the silver standard data. The following research questions shall be answered in this thesis:

1. In terms of the accuracy of text zoning label projection, to what extent do the performances of the word alignment approach and the zone tagger approach differ from each other?
2. For multilingual zone taggers, will the different types of training material play a role here, i.e., training on the unit of whole job advertisements or sentences?
3. Which foundational models are better, i.e., can the superiority of word embeddings from different language models be observed (BERT versus XLM-RoBERTa)?
4. Will 2-phase training, i.e., to fine-tune multilingual zone taggers with monolingual data, deliver improved results?
5. What are the particular characteristics of the model predictions on the test set?

1.7 Thesis Structure

The chapters that follow this introduction of the thesis are organized as follows: Chapter 2 provides an overview of the relevant literature in regards to text zoning, machine translation, cross-lingual annotation projection, and the steps involved in training multilingual zone taggers. In Chapter 3, the characteristics of the data utilized for this project, the formation of silver standard data and the rationale behind the data representation are discussed. Furthermore, Chapter 4 provides an elucidation of the methods applied for this research, which include word alignment and sequence labeling model training approaches. In Chapter 5, experiments, results, and discussions are presented in order to provide an in-depth analysis of the research conducted to address the proposed research questions. In conclusion, the final chapter of this thesis (Chapter 6) summarises the results obtained from the experiments and proposes potential directions for future research.

2 Related Work

This chapter introduces the related work of the research, which is organized into four topics in a bottom-up fashion, i.e., text zoning, machine translation, cross-lingual translation projection, and sequence labeling model training. Section 2.1 presents the previous research related to the cross-lingual annotation project, which is the core idea on which the approaches in this project are based. Section 2.2 gives an overview of the machine translation technology with a focus on the translation engine powering the creation of silver standard data for this project as well as the word alignment work, which also set the foundation of the approaches. Section 2.3 gives a summary of the previous work regarding the model training for text zoning taggers, especially the improvements brought by recent studies. Lastly, section 2.4 talks about the sequence labeling model training and other machine learning aspects related to the experiments of this project.

2.1 Cross-lingual Annotation Projection

Annotation projection between languages shares the core idea of cross-lingual transfer. The underlying principle is that annotations available for a text in one language can be projected, thanks to the alignment or other techniques, to the corresponding text in another, creating hereby a newly annotated corpus for a new language. With the development of computational linguistics, natural language processing (NLP), and the border application of language technology driven by annotated corpus, cross-lingual annotation projection is receiving more interest from academics and practitioners.

2.1.1 Text Span Annotation Projection

The early implementation of projection of annotations across languages dates back to Yarowsky and Ngai (2001), where the researchers have shown that automatically word-aligned bilingual corpora can be used to induce part-of-speech taggers and noun-phrase bracketers successfully. Since then, many studies have reported progress on transfer cross-lingual tags, especially NER tags. Ehrmann et al. (2011) automatically annotated the English version of a multi-parallel corpus and projected the annotations into other language versions. They incrementally applied different methods for the projection: perfect string matching, perfect consonant signature matching, and edit distance similarity. Furthermore, Weber and Steedman (2021) reported more recent experiments on fine-grained entity typing and showed that the previous method, which involves generating training data without manual annotation (Yarowsky and Ngai, 2001), outperformed by zero-shot cross-lingual transfer building upon XLM-RoBERTa. The task of *fine-grained entity typing (FET)* is to assign a semantic label to a span in a text. In addition, Sluyter-Gäthje et al. (2020) projected more complex structures when dealing with *shallow discourse parsing (SDP)*, which refers to the identification of coherence relations between text spans. The aforementioned text span annotation projection shares a similarity with the text zoning task since text zones also cover a wide range of text spans.

2.1.2 Automatic Markup Transfer in Translation

A recent research deals with the problem of automatic markup transfer in translation (Zenkel et al., 2021), which involves placing markup tags from a source sentence into a fixed target translation. The authors improved an algorithm (Hanneman and Dinu, 2020) for markup transfer via word alignments and proposed a supervised approach to markup transfer, which benefits from word alignments. In addition, the study introduced two novel metrics for comparing approaches to bilingual markup transfer. Similar work has been done, which focuses on the problem of simultaneous translation and markup for the fully automatic use case by Hashimoto et al. (2019). The proposed mechanisms for markup transfer shed light on this project since the text zoning shares similar characteristics to the markups. As a matter of fact, the way of thinking and design of experiments give inspiration for the primary two types of approaches carried out in the experiment of this project, which is further discussed in Chapter 4. To assess the performance of a multilingual transfer approach, some research engages in building a multilingual corpus for training and evaluation. Conneau et al. (2018) developed an evaluation set for cross-lingual language understanding (XLU) by extending the development and test sets of the Multi-Genre Natural Language Inference Corpus (MultiNLI) to 15 languages, including low-resource languages. In addition, several baselines for multilingual sentence understanding were provided, with the best performance resulting from directly translating the test data. The evaluation suite is considered to be a practical and challenging evaluation task for natural language processing systems. As for the scope of this project, the accuracy of text zone labels is the main factor in the assessment of implemented approaches.

2.2 Machine Translation

Machine translation is the vital workhorse to create the silver standard data for training zone taggers for non-German language. The quality of translation and the ability to handle XML tags of the neural machine translation system together empower the research pipeline of this project. Additionally, the word alignment benefits from neural architectures such as Transformers, which makes the straightforward approach *align-and-project* possible.

2.2.1 Neural Machine Translation

Neural machine translation, or NMT for short, is the use of neural network models to learn a statistical model for machine translation. Based on the initial Encoder-Decoder Model, NMT has been progressing quickly, particularly with the advancement of neural architectures such as Transformers (Vaswani et al., 2017). NMT has achieved state-of-the-art performance on various language pairs, and in practice, NMT also becomes the key technology behind many commercial MT systems (Tan et al., 2020). The Transformer architecture is based on a concept called attention, and more specifically, the self-attention mechanism, which facilitates the emergence of large-scale pre-trained models like BERT (Devlin et al., 2019). Transformers have enabled models with higher capacity, and pre-training has expedited their use in all types of NLP tasks. Recent transformer-based language models, such as BERT, and XLM-RoBERTa (Conneau et al., 2020), have shown a powerful ability to learn universal language representations. As for the production of silver standard data, the commercial translation service DeepL empowers the creation process. DeepL uses proprietary algorithms based on neural networks with significant differences in the topology compared to Transformer architecture, which leads to an overall significant improvement in translation quality over the public research state-of-the-art[1]. With provided off-the-shelf Python API, it is achievable to automatically translate a large amount of training data into 3 different languages in a reasonable time (79 Million characters). Furthermore, the DeepL translation engine can handle XML tags properly, which is another key point for this project.

[1] `https://www.deepl.com/en/blog/how-does-deepl-work`

2.2.2 XML Markup Language

XML tags form the foundation of XML, and they define the scope of an element in XML (Bray et al., 2008). XML tags can also be used to insert comments, declare settings required for parsing the environment, and, most importantly, encode annotation information for text spans. Right after the foundation of XML 1.0, there was already a proposal to adopt XML for data interchange between databases and other sources of data in the area of bioinformatics (Achard et al., 2001). However, to exchange information encoded with XML tags between languages, the machine translation system should have the ability to correctly transfer project XML tags. Prior to Hanneman and Dinu (2020), Müller (2017) provided a comprehensive survey of existing markup handling solutions and reimplementations of existing and novel solutions in terms of phrase-based, statistical machine translation. As in this work, DeepL API is equipped with the ability to handle XML tags. However, the technical details remain unknown[2].

2.2.3 Word Alignment

Regarding the technique of the approaches based on the alignment, *word alignment* was exploited to project the English annotations of coherence relations on the German target text and produced a German corpus with annotation accordingly. Concerning word alignment, Li et al. (2019) proposed that neural machine translation (NMT) may fail to capture word alignment through its attention mechanism to some extent, despite prior research suggesting affirmative (Bahdanau et al., 2015). They further introduced two better word alignment methods which are general and agnostic to specific NMT models: alignment by explicit alignment model and alignment by prediction difference. Word alignment naturally plays an essential role in the approaches of cross-lingual transfer. This project utilized two widely used word aligners to pipelines for comparison. `fast_align`[3] is a simple, fast, unsupervised statistical word aligner, essentially a Reparameterization of IBM Model 2 (Dyer et al., 2013). The neural aligner `awesome-align`[4] is a tool that can extract word alignments from multilingual BERT and allows model fine-tuning on parallel corpora for better alignment quality (Dou and Neubig, 2021).

[2]When asked about the technical details behind XML tag handling via Email, the support team from DeepL replied with "We do appreciate your interest a lot, however, we don't share any information publicly as the industry we are in is highly competitive and revealing technical information how our AI works would undermine our business model"

[3]`https://github.com/clab/fast_align`

[4]`https://github.com/neulab/awesome-align`

2.3 Text Zoning

The study of text zoning for job advertisements aims to partially substitute manual annotation by automatic data processing with supervised machine learning to lower data collection costs and enlarge the research span. *Text zoning* refers to segmenting the job advertisement text into zones (or classes) differing from each other regarding their content (Gnehm, 2018). Previous studies leverage mainly the off-the-shelf annotated corpus of job advertisements from the Swiss Job Market Monitor (SJMM). Purposed approaches regarding texting zoning pipeline include BiLSTM models for sequence labeling and task-specific word embeddings and ensemble techniques, which are subsequently improved by contextualized in-domain embeddings with BiLSTM-CRF models and a multi-tasking BERT model (Gnehm, 2018; Gnehm and Clematide, 2020). Other than the token-level sequence labeling task, multilingual approaches are also required since the data from SJMM are in German, French, English, and Italian. Gnehm and Clematide (2020) suggests transfer approaches, which enlightens the objective of this work.

Furthermore, the most recent work experiments with transfer learning and domain adaptation on the basis of SJMM corpus in German, whose contribution consists in building language models which are adapted to the domain of job advertisements and their assessment of a broad range of machine learning problems (Gnehm et al., 2022). Their findings show the large value of domain adaptation in terms of model performance, data shifting, and model efficiency. This work is appreciated since it provides the latest benchmark of zone tagger, which helps to evaluate the model performance from the experiments not only on German data but also on translated English, French and Italian data.

2.4 Sequence Labeling Model Training

Essentially, text zoning is a sequence labeling task. Sequence labeling has been one of the most discussed topics in computational linguistics history. Named entity recognition (NER) is probably one of the most researched sequence labeling tasks, which is tagging entities in text with their corresponding type. This project uses the `FLAIR`[5] python library to train sequence labeling models, a simple but versatile framework for state-of-the-art NLP (Akbik et al., 2019). Flair allows to use and combine different word embeddings, such as BERT and XLM-RoBERTa embeddings, in the experiments. Flair also builds directly on PyTorch and is compatible with the

[5] `https://github.com/flairNLP/flair`

HuggingFace[6] library to utilize GPU and a wide range of pre-trained models (Wolf et al., 2020). Additionally, the training pipeline implemented the techniques proposed as FLERT, which is to document-level features for sequence by defining context windows for sentences (Schweter and Akbik, 2020). The training pipeline also includes 2-phase training, which is to fine-tune multilingual models on monolingual data for potentially better performance. Similar work has been done for neural machine translation systems by generating large synthetic parallel data from minimal monolingual data in a specific domain (Marie and Fujita, 2021). Another contribution to approaches of model training discusses a technique called domain-adaptive fine-tuning which adapts contextualized word embeddings to a target domain that may differ substantially from the pretraining corpus (Han and Eisenstein, 2019). This approach was tested on two challenging domains, Early Modern English and Twitter, and it offered substantial improvements over strong BERT baselines, particularly for out-of-vocabulary words. Therefore, domain-adaptive fine-tuning is a simple and effective method for adapting sequence labeling to new domains without the need for labeled data.

[6] `https://huggingface.co/models`

3 Data

This chapter presents the description of the data used in this project, the process of data format, and the way of thinking behind the chosen data representation. Section 3.1 gives an introduction to the original source data in German, which is referred to as the gold standard data for this project. Section 3.2 provides a detailed view of the creation and processing of the multilingual silver standard data based on the original data, which is the foundation of trained machine learning models. Furthermore, section 3.3 presents the creation of a multilingual gold standard test set.

3.1 Source Data (Gold Standard)

The multilingual corpus from the Swiss Job Market Monitor[1] (SJMM) contributes to this project. SJMM arose from a research project on the long-term development of job advertisements in the press since 1950, conducted in the framework of the Swiss National Science Foundation[2] research program "Zukunft Schweiz" ("Future Switzerland"). After continuous expansion since 2002, the project transformed into a continuous scientific monitor of the job market, incorporating the internet in the modern days. The purpose of SJMM is to extract information from job advertisements to monitor and analyze trends in the Swiss job market, which benefits companies, the working population, and policymakers via well-founded information on the development of the job market.

The multilingual corpus consists of print and online job advertisements in German, French, English, and Italian. It covers the time span from 1950 up to today. For all job advertisements, high-quality human annotations of profession, industry, and management functions are available. The annotated corpus provides a great resource for training machine learning models in terms of sequence labeling. Other than in German corpus, the annotations for corpus in other languages are not complete and

[1] https://www.stellenmarktmonitor.uzh.ch/en.html

[2] https://www.snf.ch/en

Zone	Definition
z1	company description
z2	reason of vacancy
z3	administration & residual text
z4	job agency description
z5	material incentives
z6	job description
z7	required hard skills
z8	required personality (soft skills)

Table 1: Text Zones and Definitions

Dataset	Number of Job Ads	Number of Lines (Token Entries)
training	23,014	2,859,733
development	672	138,960
test	626	131,537
total	24,312	3,130,230

Table 2: Statistics of Source Data

lacking text zoning information. An important processing step in the information extraction pipelines is the token-based segmentation of the text of job ads into domain-specific text zones. As mentioned in section 1.5, text zones are defined as segmenting the job advertisement text into zones (or classes) differing from each other regarding their content. There are eight zones annotated in the corpus, and table 1 shows the label of zones and the corresponding definitions.

For the scope of this work, the newly processed gold standard data from the work of Gnehm and Clematide (2020) are used as source data to create silver standard data. In this data, text zones are annotated on German job advertisements from 1970 to 2021. This labeled data serves the purpose of supervised machine learning experiments for the texting zoning and classification tasks. The data is already split into training, development, and test set. All data sets contain 3.1 million lines of tokens and 24.3 thousand job advertisements (according to unique job advertisement IDs). Table 2 shows the statics of the dataset. In addition, there are also a number of long job advertisements truncated in order to fit into memory when training. The truncated job advertisement IDs are recorded for further inspection.

The source data represents each token and its corresponding POS tags, relative position, text zone label, and Job Advertisement ID in tabulator-separated lines, and each line is separated by line breaks. The POS tags are generated by spaCy models from German trained on TIGER corpus[3], following the scheme of the STTS (Stuttgart/Tübinger Tagset)[4]. The following clipped example shows the original data format. This format shares great similarities with the BIO format mainly used for NER tasks; for example, in CoNLL-03 shared task, hence the data can be easily adapted to the tools and machine learning code libraries for NER tasks. Other than the aforementioned, the source data are also utilized as the monolingual data used for domain adaption experiments. The beginning of this master thesis project involves the pipeline of data representation and the creation of silver standard data, which is elaborated on in the following sections.

```
[...]
Baudepartement	NN	1	10	12010112120002
,	$,	2	10	12010112120002
Umweltdepartement	NN	4	10	12010112120002
und	KON	6	10	12010112120002
Wirtschaftsdepartement	NN	8	10	12010112120002
Ob	APPR	10	70	12010112120002
Print	NN	12	70	12010112120002
oder	KON	14	70	12010112120002
Web	NN	16	70	12010112120002
:	$.	17	70	12010112120002
[...]
```

3.2 Silver Standard Data

Obtaining a sufficient quantity of adequately labeled data is becoming an increasingly difficult challenge for machine learning, especially in the case of the zone tagger trained on SJMM data. The educational materials regarding zoning are offered in German, with much fewer resources available in French, English, and particularly

[3]https://spacy.io/models/de
[4]https://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/germantagsets/

Italian. This has a disadvantageous effect on the efficacy of text segmentation when dealing with languages that are not of Germanic origin. Reliable and comprehensive datasets in multiple languages are essential for the successful implementation and reliable assessment of tag transfer systems. However, the construction of gold standard data is a huge and time-consuming process, and hence in this work, the automatically generated silver standard data served the purpose of conducting the experiments and evaluating the approaches for tag transfer. The process of automatically generating content is carried out utilizing the application programming interfaces (APIs) provided by DeepL, which enable the user to quickly and efficiently generate fresh material. This process involves taking the silver standard data and translating it into three different languages, namely English, French, and Italian.

The generated silver standard data follows the same format as the original source data provided by SJMM, only the column which indicates the token's relative position is omitted. Naturally, the tokens are in different languages. Due to the morphological differences of each language, the tokens do not have one-to-one correspondence. The silver standard data also have the same split training, development, and test set. The Figure 1 shows the overview of the workflow to generate silver standard data. Furthermore, the following subsections elaborate on the process in detail.

3.2.1 Sentence Restoration

The previous work from Gnehm and Clematide (2020) regarded the whole job advertisement as a single input, which is not tailored to this work. Machine translation works are based mainly on the unit of sentences, and the length of a sentence affects the translation quality to some extent. Hence the first step of the silver standard data creation pipeline is to convert BIO-like vertical column format into horizontal plain text and enable sentence separation from the whole job advertisement. The early attempt used the sentence splitter ("Sentencizer")[5] from the spaCy library, a simple pipeline component to allow custom sentence boundary detection logic that does not require the dependency parse. However, the results were not ideal due to unexpected splittings, e.g., some company names will be separated from the sentence. In the end, a rule-based strategy that does not require a statistical model to be loaded was implemented.

The sentence splitting was based on the job advertisement IDs and punctuation marks which indicate sentence boundaries. The common separators, like full stops,

[5] `https://spacy.io/api/sentencizer`

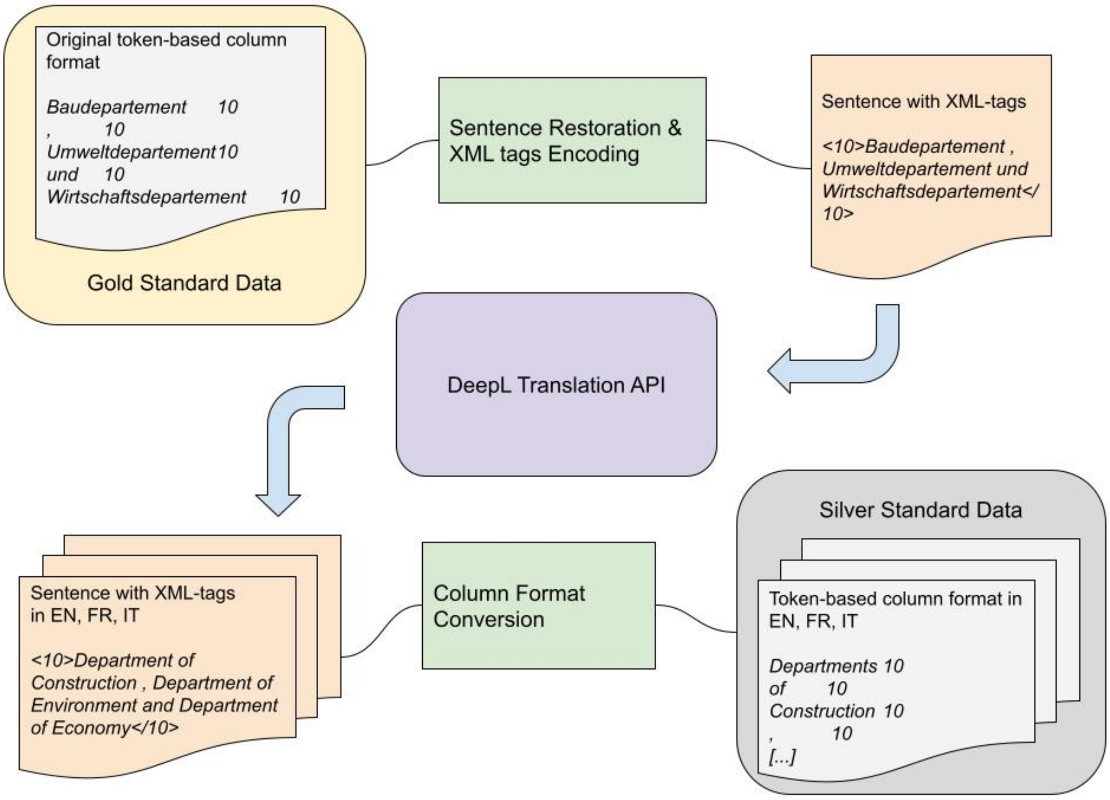

Figure 1: The Workflow of Creation of Silver Standard Data

question marks, semicolons, etc., were taken into consideration, as well as some adaptions specific to several cases, like asterisks and hyphens. This rule-based workaround has two significant benefits; the first is that the results are under control to a more considerable extent, which also leads to effective sentence length reduction, and the other is that this workaround can cooperate with the XML-tags injection process in the next step, which is essential to for the word alignment approach with preserved sentence order information. In addition, this workaround also includes several preprocessing steps for the original data.

Preprocessing of Original Data The original tokenization of the German job advertisements has split the slash within the nouns when written gender-fairly. This has led the machine translation to treat these tokens separately and output unwanted results. To tackle this problem, these gender suffixes and other gender-fair expressions such as "(in)" or "m/w" and many other cases are automatically combined with the token before, which in this way will be treated as a single token for machine translation systems. The following examples illustrate that the gender suffix in German was mistranslated into English and the alternation of the original data in the preprocessing.

The translation will keep the gender-fair suffix in German, which is not the case in English:

> *Input: Sachbearbeiter / in Customer Care Als führender Schweizer Versicherer engagiert sich die AXA Winterthur für Ihre finanzielle Sicherheit .*
>
> Translation: Clerk / in Customer Care As a leading Swiss insurer, AXA Winterthur is committed to your financial security .

The data before preprocess:

`Sachbearbeiter` `NN` `1` `60` `22011110002082`

`/` `$(` `2` `60` `22011110002082`

`in` `APPR` `3` `60` `22011110002082`

`Customer` `NE` `5` `60` `22011110002082`

`Care` `NE` `7` `60` `22011110002082`

The data after preprocess:

`Sachbearbeiter/in` `NN` `1` `60` `22011110002082`

Customer NE 5 60 22011110002082

Care NE 7 60 22011110002082

Moreover, some other "combine and append" preprocess steps were carried out manually in order to finetune translation output and avoid sentences that are too short (sentences with less than two tokens). These preprocessing steps were performed on objects such as:

- gender-fair expressions like suffixes or the ones in parentheses
- only one token after the sentence separator, usually the unique token stands for the phone number or website
- some exceptional cases, like consecutive 3 asterisks

Additionally, the hyphens in the corpus were also adjusted. Hyphens have two functions in the data: the bullet points starter and the connection of range spans such as time and numbers. To split sentences for bullet points yet keep the hyphen inside the sentence in the latter scenario, hyphens and surrounding tokens (usually cardinal numbers) were also combined as a single token. This alternation leveraged the token's POS-tag annotation in the original data.

Sentence Length Control The processing of the original data dealt with the sentences which are too short. However, sentences that are too long also need to be considered. Unlike the job advertisements in the 21st century, the ones in the early days tend to have fewer or even no full stops, which leads to very long sentences after the sentence restoration process, or sometimes the full job ads will not be split. This problem was mitigated by complementing the patterns of the sentence separator, e.g., question marks, semicolons, or asterisks and hyphens as bullet points starter. The following 2 figures illustrate the comparison from the expansion of the sentence separator list with the plot of sentence length distribution in the test set. The x-axis indicates the length of the sentence, and the y-axis indicates the count in Figure 2 and the density in Figure 3. The left subfigure shows the distribution of sentence length when only the full stop is counted as the separator (version v1 in Figure 3) while the right shows the results of the expansion (version v2 in Figure 3). The kernel density estimate (KDE) plot in Figure 3 presents clearly that the overall sentence length was drastically reduced, because of the increment of density for the sentences around the average length. It is evident that a large number of sentences exhibit lengths of between 100-300 tokens, which is substantially longer than what is conventionally expected for a sentence. Initial apprehensions of decreased quality

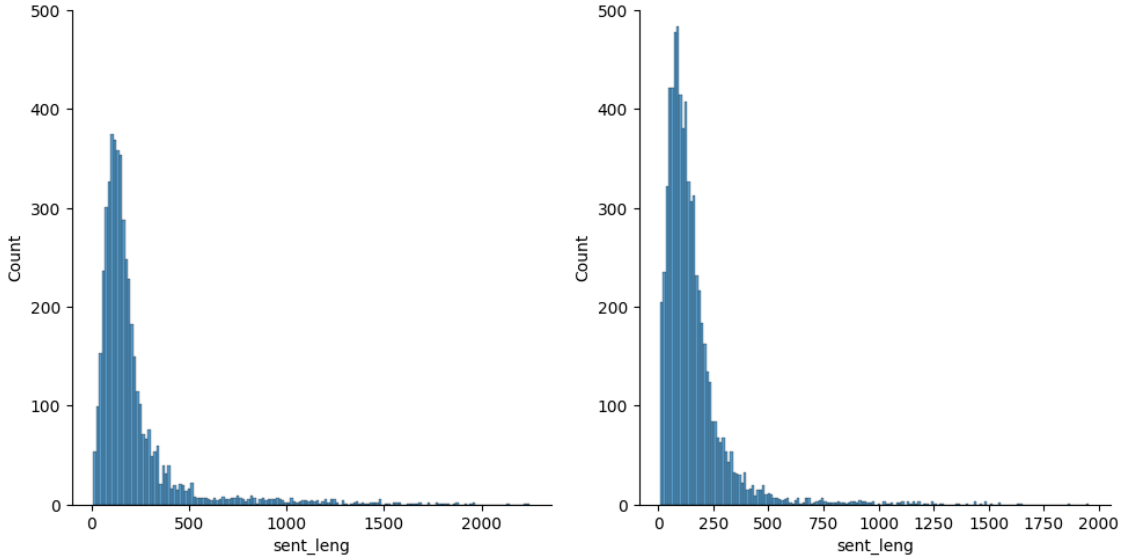

Figure 2: The Distribution of Sentence Length in Test Set

in machine translation have been refuted by more recent evidence, which suggests only a slight decrease in its effectiveness.

3.2.2 XML tags encoding

As mentioned before, XML tags are suitable to encode annotation information into text spans and represent the data in plain text, which is the only acceptable input for the machine translation system and compatible with approaches for markup transfer. Algorithm 1 shows the pseudocode developed for the addition of XML tags and the separation of sentences in parallel. Sentences without XML tags were also generated for reference, and the sentence order and numbering data were also recorded during the steps for the later word alignment. The total process of generating split sentences with XML tags takes around 30 min for the training data.

3.2.3 Translation via DeepL's API

The generated sentences with XML tags were then automatically translated by the DeepL machine translation tool, which is realized by the Python API provided by DeepL. The translation engine does not take tags into account by default unless the tag handling setting is adjusted to "xml". Moreover, the API will process XML

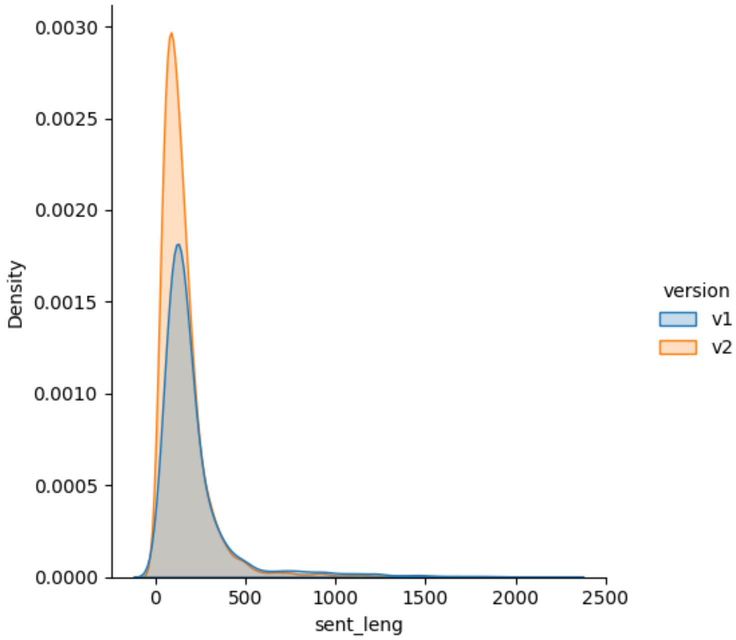

Figure 3: The KDE Plot of Distribution of Sentence Length

Algorithm 1 Sentence Restoration with XML tags

```
read original_dataset as whole_job_dataframe
get job_dataframe from whole_job_dataframe
create text_list
create text_span
define sentence_separator_list
for row in job_dataframe do
    get token, tag from row
    if tag is different from the tag in previous row then
        if token is in sentence separator list or token is the last token then
            append close_tag, new_tag, token, new_close_tag to text_span
            append text_span to text_list
        else
            append close_tag, new_tag, token to text_span
        end if
    else
        if token is in sentence_separator_list or token is the last token then
            append token, close_tag to text_span
            append text_span to text_list
        else
            append token to text_span
        end if
    end if
end for
return text_list
```

Data Set	English	French	Italian
Training Set	3,205,458	3,555,346	3,280,976
Development Set	154,828	173,051	158,444
Test Set	146,911	164,824	151,034
total	3,507,197	3,893,221	3,590,454

Table 3: Statistics of Silver Standard Data (Number of Tokens)

input by extracting the text out of the structure, splitting it into individual sentences (or text spans in this case), translating them, and placing them back into the XML structure. By the process of the algorithm, the sentences with XML tags generated from the SJMM column format will not have nested tags, and each token can be enclosed by maximally one tag pair since each token has a single text zoning label. Furthermore, many sentences only contain one tag pair, i.e., the whole sentence is enclosed by the open tag at the beginning and the closing tag at the end. Overall, in the test set, 43.5% of total sentences contain more than one tag pair, in the development set: 42.4%, and in the training set: 53.4%.

For the scope of this master thesis project, the original data was translated into 3 languages: American English (EN), French (FR), and Italian (IT). The average translation speed via DeepL API is 6 sentences per second. For training sets FR and EN, they cost 12 hours each, while translating the IT training set took 20 hours. The price for DeepL API is € 20.00 per 1 million characters. The character usage of the translation process is 79 Million, which amounts to around € 1580 to generate the silver standard data.

3.2.4 Column Format Conversion

The translated sentences with XML tags were then further processed to convert into the column format to match the original data. The tags were removed to get plain text in order to create parallel text data for word aligners. The conversion is then carried out with several twerks, e.g. recovering the space before the full stop at the end of the sentence. This last step generated 3 formats of data in 3 languages, namely the token-based column format and job ads split into sentences with and without XML tags. These generated data created parallel multilingual and monolingual data for model training and fine-tuning in the following experiments. The following table 3 show the generated silver standard data statics in 3 languages.

3.2.5 Quality control

The creation of silver standard data entails no manual correction and entirely relies on the quality of the translation engine, i.e., DeepL. Even though DeepL has been proven to deliver industry-leading results, some quality control measurements were accomplished to ensure the silver standard data meets the quality requirements.

Before Translation Before the translation process began, 200 randomly selected sentences from the test and development set were translated via DeepL API in EN, FR, and IT. The translation was then thoroughly evaluated by the supervisors and author of this master's thesis from the perspective of the quality of translation and quality of tag segmentation (tag transfer). For the tag transfer, there are no issues such as missing tags or incorrect transfer. The segmentation problem is also minimal; of all languages, there are 1 or 2 cases that the segmentation needs to be manually adjusted. The problem regarding the quality of translation does exist but is minimal. Typical issues include the mistranslation of entities like company names and unique name holders and issues like gender-fair suffixed as mentioned in section 3.2.1. Additionally, the evaluations of the quality of translation in English were carried out by the supervisors and the author simultaneously and independently. The computed inter-evaluator agreement on the English samples is 0.74, which indicates that the issues in the translation are consistent. Based on these observations and the fact that the evaluation of the quality of translation is beyond the scope of this project, it is safe to conclude that the production of silver standard data meets the quality requirements for machine learning and is valid for further research. The evaluation also investigated the translation differences between the sentences with and without XML tags and the distinctions are not significant.

After Translation The quality control after the translation intends to address the issues from the translation process and the API. For some sentences (133 cases), DeepL's API returns an empty string, especially when translating from German into French. This bug from its API is possibly triggered by an asterisk or hyphen at the end of a sentence. Removing the ending punctuation in the source language and appending it after the translation was a temporal workaround. There were no other issues with the translation results regarding the API.

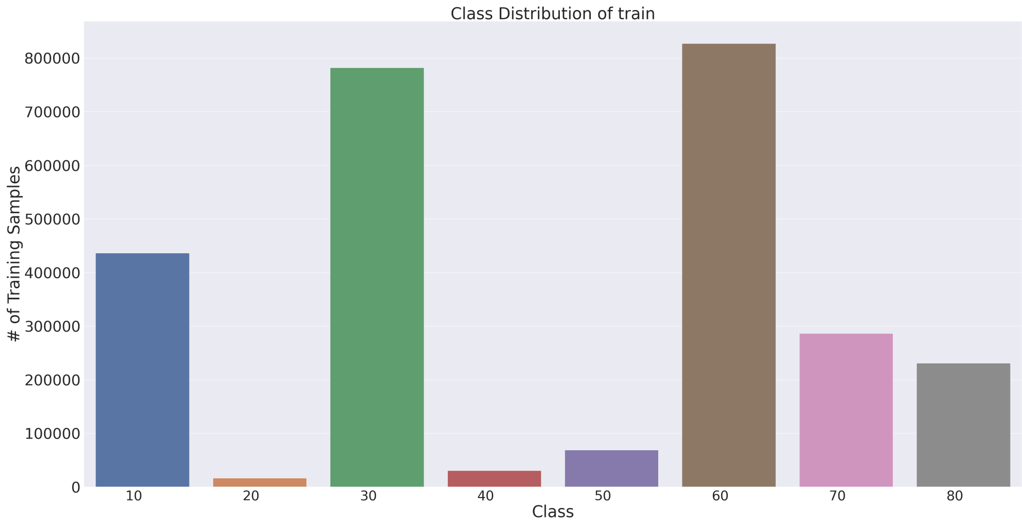

Figure 4: Label Distribution in the German Training Set

3.3 Gold Standard Test Set

We also seek to evaluate the performance of the models when tested on job advertisements that have not gone through translation, as the current silver standard only comprises of job postings that have been translated, which may result in discrepancies related to the language used and the amount of translationese included. To gain an exhaustive insight into the projection of tags between languages, a set of job postings from 2001 to 2022 that were written in English, French, and Italian were pre-classified using zone taggers that had been trained using a silver standard data set. The process of compiling the definitive, accurate set of data in three distinct languages was conducted by a single individual who manually reviewed and verified each sample. The incorporation of a gold standard evaluation test into the existing silver standard test suite is a critical component of the assessment methodology, facilitating the reliable evaluation of the data. This gold standard test set comprises a total of 23,009 token entries, including 7,357 in English, 7,643 in French, and 8,009 in Italian.

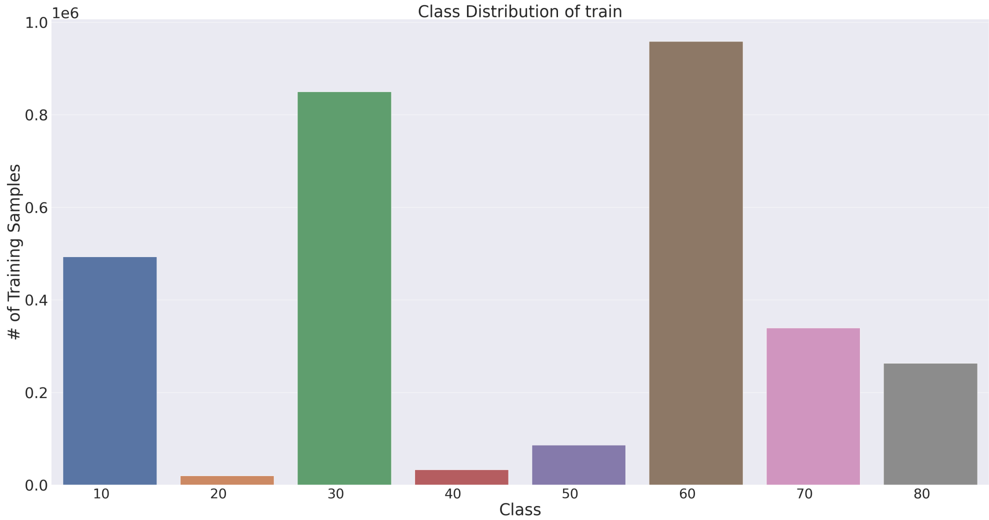

Figure 5: Label Distribution in the English Training Set

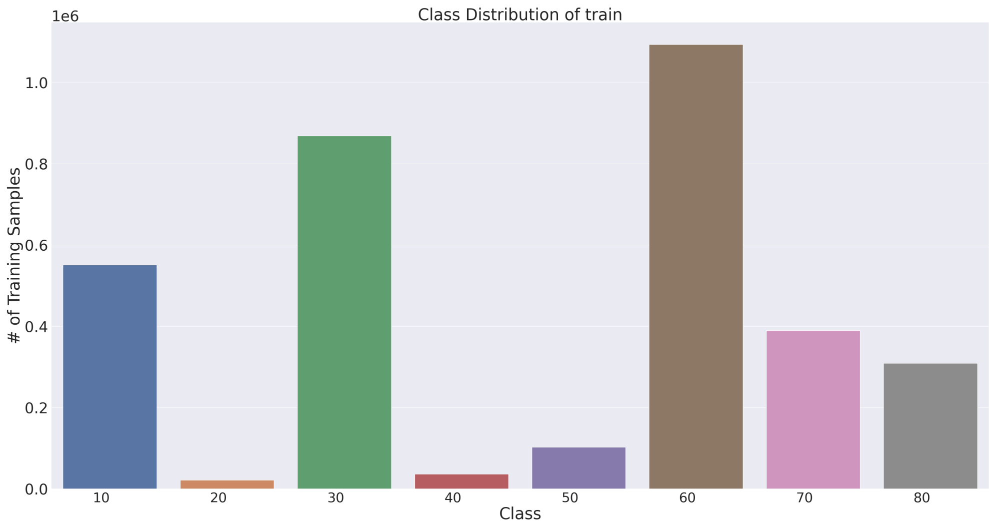

Figure 6: Label Distribution in the French Training Set

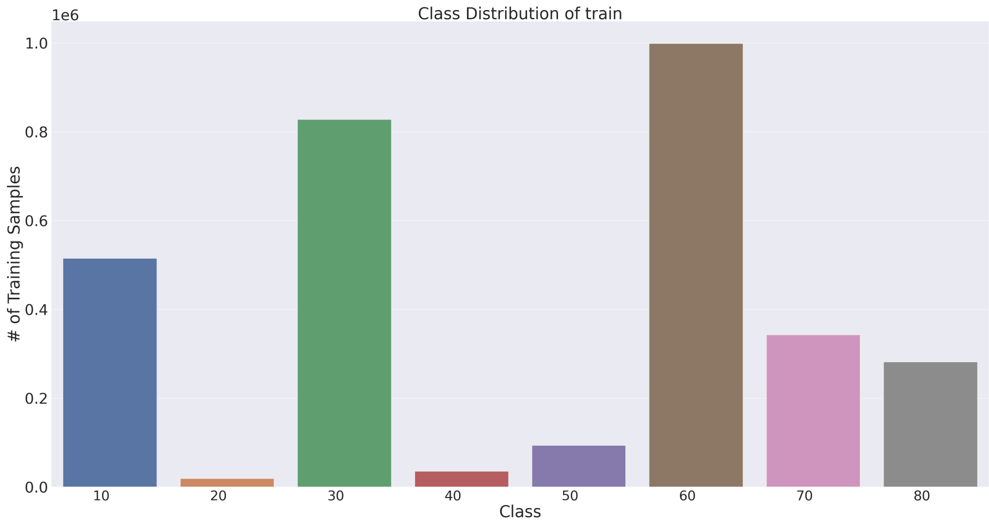

Figure 7: Label Distribution in the Italian Training Set

3.4 Data Imbalance

Unbalanced datasets are a widespread problem in the fields of machine learning and data science. When there is an imbalance between the number of samples from different classes, resulting in an unequal distribution of data between categories, this is known as imbalanced data. This disproportion can lead to a lack of precision in machine learning algorithms as a consequence of the predominance of the most frequent category. In order to deal with the unequal distribution of data, numerous strategies can be implemented. Such as increasing or decreasing the frequency of the scarce classes, using different metrics for measuring performance, or utilizing algorithms specifically designed to handle this type of data. In addition, data augmentation can be employed to generate more examples of infrequent categories, thus evening out the distribution of the dataset. The below diagrams demonstrate the division of text zoning classifications present in the training set for a variety of languages. Figures 4, 5, 6, and 7 show the distribution of the gold standard German training set as well as the silver standard training sets in English, French, and Italian, respectively. The data collected in the training set reveals that text zones 60, 30, and 10 are the most common, while text zones 50, 40, and 20 are scarcely represented. The full resolution of the data imbalance issue is not part of the focus of this project; nevertheless, the effects of the data imbalance are carefully assessed with the utilization of the confusion matrix plot in Chapter 5.

4 Methods

This chapter introduces the methods employed in this work for the experiments presented below. The first type of experiments can be grouped as the align-and-project approach, which is to align tokens between the source and translated target language and project the labels of the manually annotated source corpus along the word alignments. The second type of methods is to train the multilingual zone taggers on the created silver standard data. The following sections elaborate on each type of methods in detail.

4.1 Align And Project

The *align-and-project* approach directly copes with cross-lingual annotation projection tasks. With available translated parallel corpus in the source and target language, as well as the annotation information in the source language, word aligners can produce the alignment information for the tokens in each sentence. Alignment algorithms can leverage this information to project annotation from the source language into the target language. Figure 8 illustrates the pipeline of the align-and-project approach with examples in German and English.

Word aligners take tokenized parallel sentence pairs of source and target language as input, where sentences in source and target languages are separated by a triple pipe symbol with leading and trailing white space. In this work, sentences in the target language can either be translated sentences from silver standard data or come from the gold test set as well. The output is the widely used i-j "Pharaoh format", where a pair i-j indicates that the i^{th} word (zero-indexed) of the left language (by convention, the source language) is aligned to the j^{th} word of the right sentence (by convention, the target language). Both statistical word aligner `fast_align` and neural word aligner `awesome-align` follow this data representation convention.

In this work, a look-up algorithm is employed to project annotations for the alignment algorithm; this algorithm retrieves the text zone label from the token in the source language and assigns the same label to the corresponding token in the target

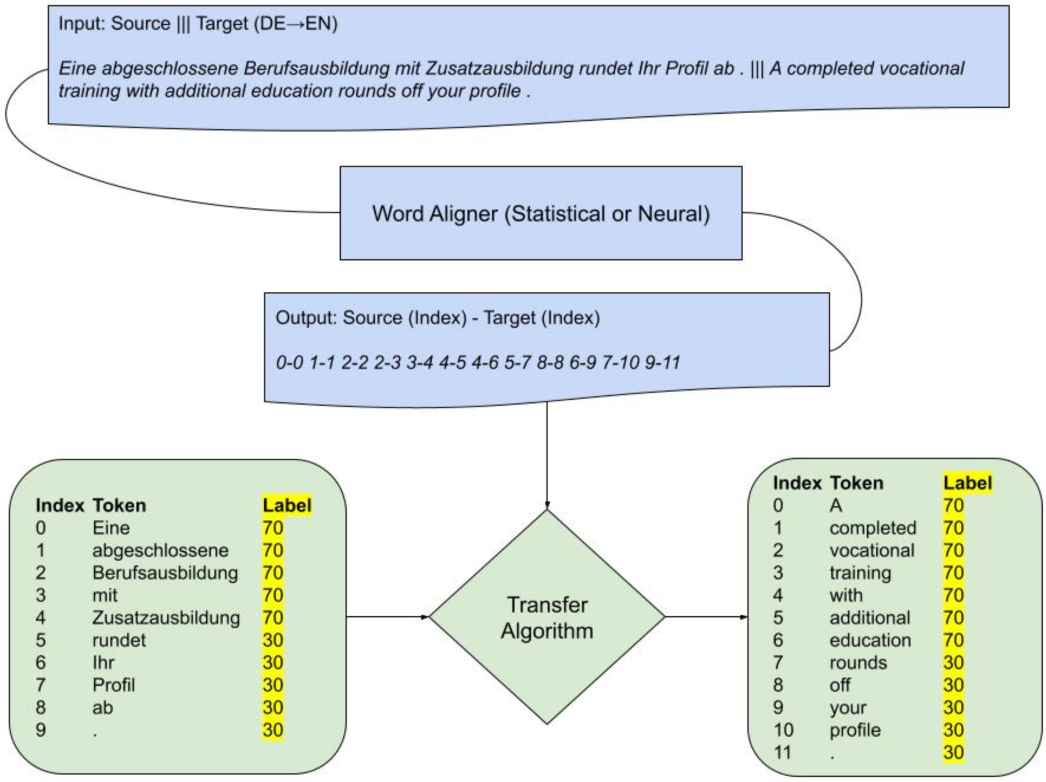

Figure 8: The Workflow of Align-and-Project Approach

language, with reference to the alignment data. The viability of this search algorithm is contingent upon the fact that every token has an unique text zone label, in addition to there being no nested labels. Occasionally, the word alignment tool will yield results with tokens that are not in alignment. It is commonly observed that misalignment of words is due to the lack of accuracy in their prediction. The post-processing for these unaligned tokens is based on two methods:

1. For each token without alignment data, the label for this token is assigned by the previous token (the same class as the one before)
2. If the token without alignment data is the first token in this sentence, then the class for this token is assigned by the first following token with available alignment data (the same class as this token)

Despite their simplicity, these methods work in uttermost cases, because the text zones are span annotation, and zone labels tend to be clustering. The following example briefly illustrates the outcome of the methods. Token spans "in" and "field of" in sentence **A** are lacking the alignment information (marked with `<unaligned>` tag pair), and the sentence **B** shows the result of the post-processing.

> **A** `<10>`[We are a leading manufacturer of corrugated cardboard packaging for industrial , food and non-food sectors and very successful`</10>` `<unaligned>`in`</unaligned><10>`the`</10><unaligned>`field of`</unaligned>` `<10>`customized packaging solutions .`</10>`
> **B** `<10>`[We are a leading manufacturer of corrugated cardboard packaging for industrial , food and non-food sectors and very successful in the field of customized packaging solutions .`</10>`

4.2 Zone Tagger Training on Silver Standard Data

Besides the approach with word alignment, the project's focal point is to train multilingual sequence labeling models, also called zone taggers, via the generated silver standard data in English, French, and Italian, as well as the gold standard train set in German. The `FLAIR` Python library[1] enables the training pipeline and model structure with reference to the previous work done by SJMM researchers. Figure 9 shows the abstraction of the training process. The multilingual training set is fed into the `FLAIR` framework, to generate Transformer word embeddings and train sequence labeling models. For this project, the word embeddings from BERT and

[1] `https://github.com/flairNLP/flair`

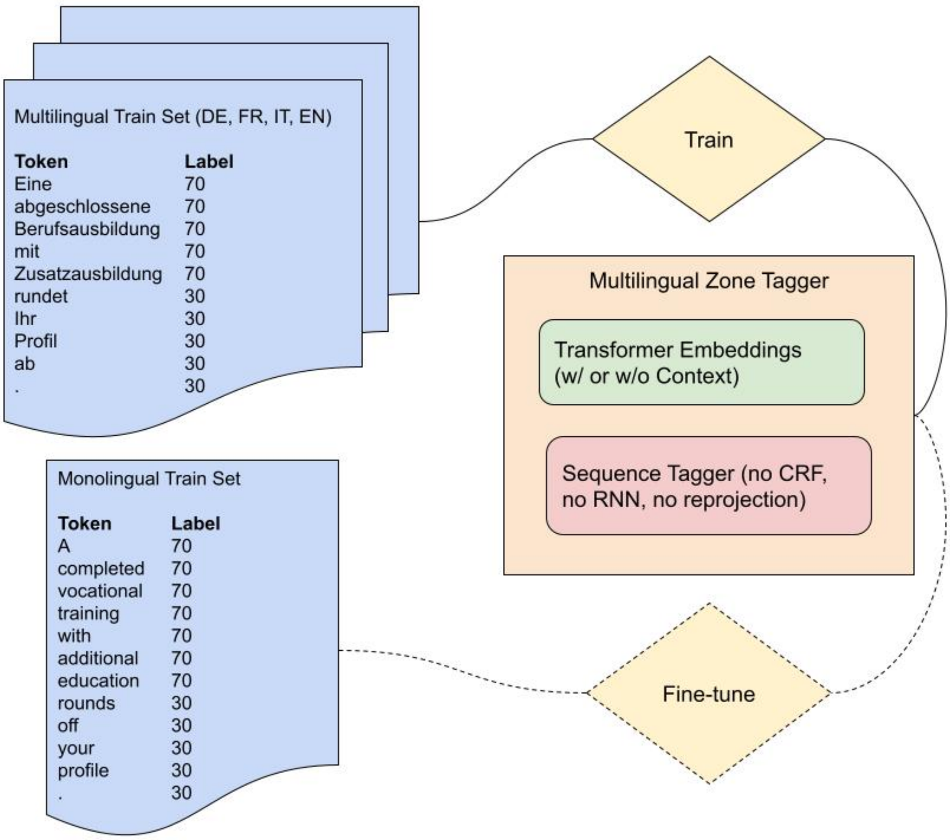

Figure 9: The Workflow of Zone Tagger Training on Silver Standard Data

XLM-RoBERTa[2] are used. To train the models, `FLAIR` includes the `ModelTrainer` class[3], which implements a host of mechanisms that are typically applied during training. This includes features such as minibatching, model checkpointing, learning rate annealing schedulers, evaluation methods, and logging.

Furthermore, during the experiments, multilingual zone taggers are either trained on the unit of whole job advertisements, or on the unit of split sentences as processed when generating silver standard data. One of the major downsides of attempting to train models on all job advertisements is that some especially lengthy job postings may be unable to fit into the allocated GPU memory. Consequently, utilizing a training model that takes contextual factors into account, such as the one provided by FLAIR's FLERT configuration, can be especially useful. Nevertheless, if the whole job advertisements are used for training the model, it can be more advantageous due to the additional contextual information it can then access.

In conjunction with the standard training routine, this project also tested methods of 2-phase training, which is to fine-tune multilingual zone tagger on monolingual data for another round, to examine if there is an enhancement of performance on the monolingual test set. This method is also known as unsupervised domain adaptation, which has gained increasing attention recently due to its potential in improving the performance of natural language processing tasks (Marie and Fujita, 2021). More specifically, unsupervised domain adaption is a type of transfer learning that allows a model trained on one domain to be applied to a different domain. This can help reduce the amount of data and time needed to train an accurate model on a new task with limited data. Unsupervised domain adaption works by first training a model on the source domain, then using that trained model as the basis for a new model in the target domain. The model is adapted using unsupervised methods, in this case, fine-tuning on monolinguial data, which allow it to learn how to generalize across domains without needing labeled data from both domains. The implementation and evaluation of 2-phase training is further discussed in Chapter 5.

[2] `https://github.com/flairNLP/flair/blob/master/resources/docs/embeddings/TRANSFORMER_EMBEDDINGS.md`

[3] `https://github.com/flairNLP/flair/blob/master/resources/docs/TUTORIAL_7_TRAINING_A_MODEL.md`

5 Experiments & Results

This chapter introduces the experiments conducted and the corresponding results for this master thesis project, as well as the discussions of the results. The experiments are grouped by the implemented approaches mentioned in Chapter 4: section 5.1 presents the experiments of the approach align and project, and the results and discussions of the performance of word aligners, and section 5.2 elaborates on the training process and results of the trained sequence labeling models. Additionally, section 5.3 talks about the outcomes in general and possible future improvements for the experiments.

5.1 Word Alignment

The experiments for the word-alignment-based approaches were realized by the derived parallel corpus from silver standard data in 3 language pairs, i.e., German to English, French, and Italian. Moreover, the performance of word aligners was evaluated on the silver test set since no human-translated and projected corpus was available for this project. The statistical word aligner `fast_align`n worked in an unsupervised fashion, therefore it could be directly applied to the silver test set and output alignment data. Based on this data and transfer algorithm introduced in Chapter 4.1, the "predictions" of word aligners in English, French and Italian could be generated and evaluated, corresponding to the test set in German. Neural word aligners, on the other hand, are mainly based on pre-trained language models, which are capable of the fine-tuning process to update model parameters.

In this work, the neural word aligner `awesome-align` was first used in the original version, which is built on the `bert-base-multilingual-cased` language model. Then fine-tuning of the `bert-base-multilingual-cased` took place on the parallel data from the silver standard training set in all 3 language pairs. As recommended by the developers of `bert-base-multilingual-cased`, the fine-tuning process was carried out with one epoch for each language to balance between efficiency and effectiveness. The fine-tuning process of 3 epochs in the model lasted 6.5 hours,

		Recall (macro average)				Accuracy			
No.	Aligner	EN	FR	IT	avg.	EN	FR	IT	avg.
1	fast_align	0.9710	0.9546	0.9498	0.9585	0.9807	0.9699	0.9660	0.9722
2	awesome-align	0.9874	**0.9751**	0.9729	0.9785	0.9902	**0.9826**	**0.9800**	**0.9843**
3	awesome-align (fine-tuned)	**0.9910**	0.9749	**0.9735**	**0.9798**	**0.9929**	0.9819	0.9780	0.9842
	avg.	0.9832	0.9682	0.9654	0.9722	0.9879	0.9781	0.9747	0.9802

Table 4: Word Aligner Performance on Silver Test Set

resulting in a perplexity of 0.3033, 0.3128, and 0.3498 after each epoch, respectively. The raising of perplexity values after each epoch indicated that the model was overfitted to a certain extent and the machine-translated data may not bring benefits to fine-tuning word aligners in this case, which was also reflected in the model performance. Overall, the experiments for the word alignment approach were carried out by these 3 word aligners.

5.1.1 Performance of Word Aligners

Table 4 shows the performance of all 3 word aligners evaluated on the silver test set, with evaluation metrics of recall in macro average and accuracy, as well as the average values per column (per language) and row (per word aligner). The best score in each column is marked in **bold**. The results demonstrate a correlation between the performance of word aligners and scores in recall and average. The neural word aligner `awesome-align` has evidently reduced alignment error rates (AER) in German-to-English and French-to-English language pairs compared to the statistical word aligner `fast_align`. This analysis proves the superiority of `awesome-align` with higher recall and accuracy scores in each language, although to a narrow extent, especially for the test set in English. Other than English, the recall and accuracy scores have an increase in the range of 0.02 to 0.03 in French and Italian, suggesting that data in French and Italian could benefit more from neural word aligners.

Additionally, fine-tuning helps neural word aligners achieve better scores in the recall however at the cost of accuracy. The increased perplexity in the fine-tuning process implies that the language model was overfitted, hence the reduction of accuracy scores surfaced. The increment of recall in macro average indicates that the classes

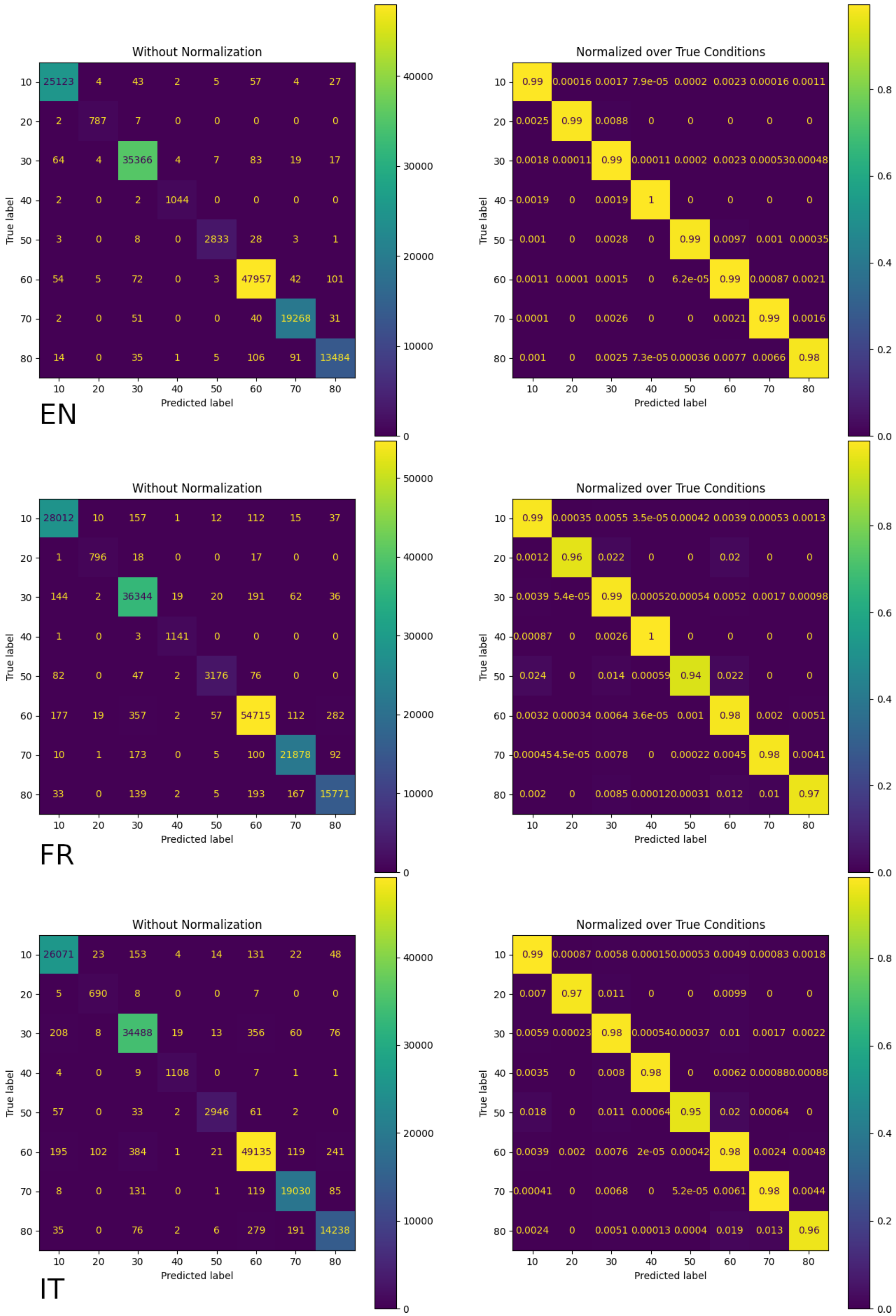

Figure 10: Confusion Matrix of Predictions of fine-tuned `awesome-align`

with fewer presences (small counts in the data set) could gain the same attention from the model as the classes in large numbers. This class imbalance is one of the characteristics of the training and testing data for this work. Figure 10 shows the confusion matrices plotted by the predictions of fine-tuned awesome-align (No.3 in table 4) in English, French and Italian. For each figure, the subplot on the left shows the values without normalization (absolute numbers), and the right subplot shows the values normalized over the true condition (values add up to 1 in each row). The figures show that all classes have a high recall score (mostly above 0.97) in each language, regardless of the imbalance in class distribution.

5.1.2 Results Discussion

In general, all three word aligners, regardless of their type of mechanics, delivered exceptional results. Despite ranking at the bottom of the list, the statistical word aligner `fast_align` achieved an average accuracy of 0.97, and neural word aligners improved this score by 0.01. However, the reliability of these data is impacted by the fact that the test set is entirely automatically generated by machine translation systems. The word aligner may have a substantial advantage on the machine-translated data due to the fact that they share many essential technical underpinnings with machine translation systems. The neural machine translation and neural word alignment are both based on language models, and statistical word alignment is built on top of statistical machine translation as well. Furthermore, the translated text tends to have less variety in terms of lexical, which could also contribute to the performance of word aligners.

Due to the lack of data on actual human-translated and projected gold test sets, the results cannot confirm that the approaches based on word alignment have a significant advantage over the other approach, which is the training of sequence labeling models (zone taggers). One drawback of the word aligners is that they can only work on parallel data in desired language pairs. The parallel data are not always readily available and, in many cases, are totally out of reach. The experiments show the potential of the align-and-project approaches, yet further research is needed to establish reliable evaluation test methods to fully assess the capability of these approaches. In addition, the complete statistics and plots of word aligners can be found in Appendix A.

5.2 Trained Models

The experiments for the sequence labeling training were realized by the created silver standard data based on the machine translated text with XML tags in 3 languages, i.e., English, French, and Italian, as mentioned in Chapter 3. In terms of the training process, Python library `FLAIR` is the main power horse, as mentioned in Chapter 4. In addition, several pretrained language models, from both the generic domain and the research team of SJMM, were the initialization of word embeddings, and they also provided a valuable baseline for model evaluation. The following subsection 5.2.1 introduces the training process in detail, while the subsection 5.2.2 elaborated the results of trained models as well as analysis about them. Furthermore, subsection 5.2.3 provides some error analysis with concrete examples from the model predicted data.

5.2.1 Training Process

Table 5 gives an overview of the training details from all 9 trained or fine-tuned sequence labeling models (zone taggers). The first column indicates the numbering of models, which is for reference, consistent in this chapter, either in main texts or in tables. The second column indicates the word embedding each that the model is based on. Decimal numbers mean that the models are fine-tuned in monolingual training data on the basis of previous trained multilingual models, or in other words, via the 2-phase training process. Since the 2-phase training was carried out in 4 languages researched in this project, there are 2 model groups with 4 decimal numbers each. Furthermore, models differ from each other with mainly the type of training data, which is either based on whole job advertisements or on the splitted sentences by rule-based pipelines. If models were trained on the sentence-based data, they can be further distinguished by the application of context, which is a mechanism proposed by `FLERT`.

Except for the first model with 20 training epochs, the epochs for other models are set to 10. Minibatch sizes were adjusted for each model accordingly, and the training time mainly depended on the batch size and type of GPU. For this work, two GPUs were utilized, the first is the Nvidia Tesla T4 with 16 GB video memory, and the second is Nvidia RTX 3090 with 24 GB video memory. The other hyperparameter for training was set identical with the default `FLAIR` settings, such as `AdamW`[1] is

[1]`https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html`

No.	Embedding `model-alias-for-reference`	Training Data	Context	2-phrase	Language	max_ epochs	Mini Batch Size	Training Time (Hours)
1	bert-base-multilingual `bert-base-multilingual-cased`	job ads	no	no	Multi	20	32	25
2	bert-base-multilingual `bert-base-multilingual-cased_w_context`	sentences	yes	no	Multi	10	16	42
3.1	bert-base-multilingual `bert-base-multilingual-cased_2_DE`	job ads	no	yes	DE	10	32	3
3.2	bert-base-multilingual `bert-base-multilingual-cased_2_EN-US`	job ads	no	yes	EN	10	32	3
3.3	bert-base-multilingual `bert-base-multilingual-cased_2_FR`	job ads	no	yes	FR	10	32	3
3.4	bert-base-multilingual `bert-base-multilingual-cased_2_IT`	job ads	no	yes	IT	10	32	3
4	jobBERT-de `jobad_bert_finetune_multi`	job ads	no	yes	Multi	10	32	13
5	xlm-roberta-base `xlm-roberta-base_w_context`	sentences	yes	no	Multi	10	16	44
6	xlm-roberta-base `xlm-roberta-base_o_context`	sentences	no	no	Multi	10	8	62
7.1	xlm-roberta-base `xlm-roberta-base_w_context_2_DE_sents`	sentences	yes	yes	DE	10	16	11
7.2	xlm-roberta-base `xlm-roberta-base_w_context_2_EN-US_sents`	sentences	yes	yes	EN	10	16	10
7.3	xlm-roberta-base `xlm-roberta-base_w_context_2_FR_sents`	sentences	yes	yes	FR	10	16	11
7.4	xlm-roberta-base `xlm-roberta-base_w_context_2_IT_sents`	sentences	yes	yes	IT	10	16	10
8	xlm-roberta-base `xlm-roberta-base_o_context_job`	job ads	no	no	Multi	10	16	15
9	"jobadBERT-multi" v2021-10-18 epoch 30 `xlm-roberta-base-job`	job ads	no	no	Multi	10	16	14

Table 5: Detail of Trained Models

Figure 11: Loss Plot of `bert-base-multilingual-cased`

optimizer, and training rate scheduler `OneCycleLR`[2], and the initial learning rate for all models is 0.000005 (5.0e-6). In terms of model structure, the hidden size was set as 256 for all models. All the models were based on the Transformer embeddings, hence there are other 2 settings for embeddings identical among models. The layer parameter was set to -1, which means only the last layer is used. Additionally, since the Transformer-based models use subword tokenization, the subtoken pooling was set to "first", which means only the embedding of the first subword is used.

Model 1-3 were based on the `bert-base-multilingual-cased`, which is by far the most widely-used language model for multilingual settings. Model 1 was trained on the basis of whole job advertisements, while model 2 was on the basis of sentences. It is worth mentioning that the gold test set was based the predictions of model 1, followed by the human correcting process, cf. Chapter 3.3. Model 3.1 to 3.4 were based on the model 1, with an extra 2-phase training process. Model 5-8 were based on the `xlm-roberta-base` in a similar manner. XLM-RoBERTa is a multilingual version of RoBERTa. It is pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages. Model 4 and 9, however, were not based on the general domain language models. Model 4 was the fine-tine product of zone tagger based on `jobBERT-de`[3]. `jobBERT-de` is based on `bert-base-german-cased` and adapted

[2]`https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.OneCycleLR.html`

[3]`https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html`

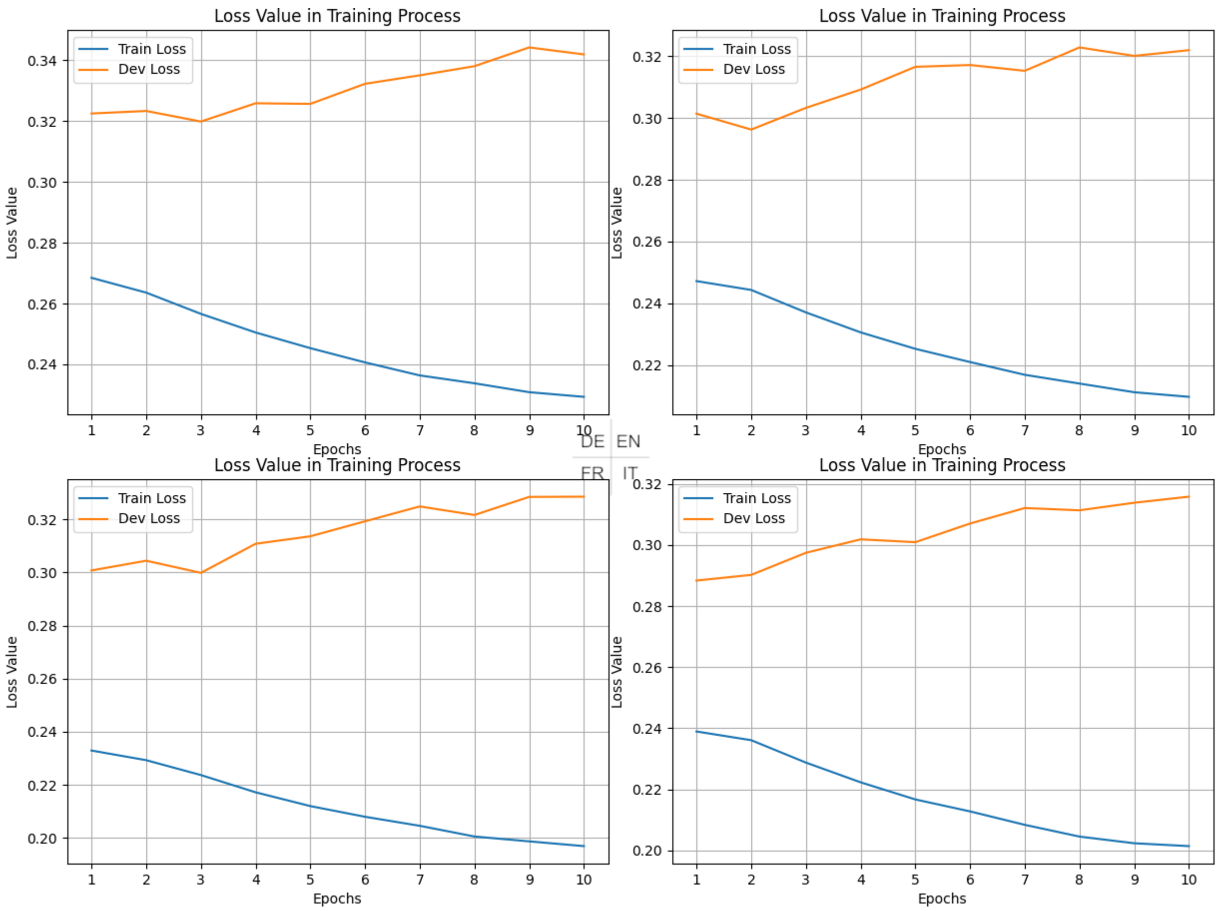

Figure 12: Loss Plot of Models via 2-phase Training

to the domain of job advertisements through continued in-domain pretraining on 4 million German-speaking job advertisements from Switzerland in the time span of 1990-2020 (5.9 GB data). While model 9 was trained as a zone tagger from the ground up, which was based on the `xlm-roberta-base` model with continued masked language model training on English, French, German and Italian job advertisements. The epoch 30 of the pretraining checkpoint was used for further fine-tune procedures.

As the first model trained, Model 1 not only contributed to the creation of the gold test set, but also provided a valuable reference for the training settings of the following models. Figure 11 shows the plot of training loss (blue line) and validation loss (yellow line) over the training epochs. According to the figure, training loss had a drastic decrease in the first 2 epochs, and started to fall off gradually. The validation loss started to rise up after the 4th epochs, implying the model was moderately overfitted, and the line of validation loss continued to increase until the intersection with the training loss after the 8th epoch. This gave the idea that the model had definitely been overfitted with 10 epochs, hence the following models were all trained with the setting of maximal 10 epochs. Figure Figure 12 shows the training loss and validation loss plot of model 3.1 to 3.4, which are annotations of monolingual settings in each language. The validation line in the plots tends to move up, which supports the findings from the training process of model 1, indicating that models were already overfitted and 10 epochs of training was sufficient. It was the same case for the rest of the models, and all the detailed training status and plots of all 9 trained models can be found in appendix B.

5.2.2 Results Discussion

Since the training data was categorized into two types: the whole job advertisement based and sentence based, intuitively, the test set could also be categorized into these two types. This raised the question on which type of test set the models should be evaluated. To clarify this point, all models were firstly evaluated on both types of test set. Table 6 presents of the model performance on both types of test set. The results show that 8 of 9 models achieved better scores on the test set based on whole job advertisements, compared to the test set based on the sentences, regardless of the type of training data. For example, `xlm-roberta-based` model 6 was trained on the sentences data without context information. Yet it achieved an average accuracy of 0.9213 on the silver test set of whole job advertisements, compared to 0.9199 on the test set of sentences. The only contradiction is Model 2, which is trained on sentences and had better scores on sentence-based test sets.

		Test Set Type	Silver Test Set			
No.	Model Description		EN	FR	IT	avg.
1	bert-base-multilingual (job ads, w/o context)	**Job ads**	0.9218	0.9244	0.9214	0.9225
1	bert-base-multilingual (job ads, w/o context)	Sentences	0.8667	0.8645	0.8627	0.8646
2	bert-base-multilingual (sentences, w/ context)	Job ads	0.9176	0.9243	0.9218	0.9212
2	bert-base-multilingual (sentences, w/ context)	**Sentences**	0.9196	0.9252	0.9238	0.9229
3	bert-base-multilingual (job ads, 2-phase)	**Job ads**	0.9229	0.9255	0.9235	0.9240
3	bert-base-multilingual (job ads, 2-phase)	Sentences	0.8625	0.8645	0.8619	0.8630
4	bert-jobad (job ads, w/o context, 2-phase)	**Job ads**	0.9111	0.9054	0.9049	0.9071
4	bert-jobad (job ads, w/o context, 2-phase)	Sentences	0.8423	0.8093	0.8266	0.8261
5	xlm-roberta-base (sentences, w/ context)	**Job ads**	0.9235	0.9308	0.9291	0.9278
5	xlm-roberta-base (sentences, w/ context)	Sentences	0.9226	0.9276	0.9278	0.9260
6	xlm-roberta-base (sentences, w/o context)	**Job ads**	0.9178	0.9236	0.9225	0.9213
6	xlm-roberta-base (sentences, w/o context)	Sentences	0.9158	0.9225	0.9215	0.9199
7	xlm-roberta-base (sentences, w/ context, 2-phase)	**Job ads**	0.9235	0.9308	0.9300	0.9281
7	xlm-roberta-base (sentences, w/ context, 2-phase)	Sentences	0.9212	0.9279	0.9285	0.9259
8	xlm-roberta-base (job ads, w/o context)	**Job ads**	0.9251	0.9292	0.9291	0.9278
8	xlm-roberta-base (job ads, w/o context)	Sentences	0.8801	0.8858	0.8855	0.8838
9	xlm-roberta-jobad (job ads, w/o context, finetune)	**Job ads**	0.9243	0.9307	0.9295	0.9282
9	xlm-roberta-jobad (job ads, w/o context, finetune)	Sentences	0.8911	0.8957	0.8916	0.8928
	all average	Both	0.9064	0.9076	0.9076	0.9072

Table 6: Model Performance (2 Types of Test Set)

		Silver Test Set				Gold Test Set				
No.	Model Description	EN	FR	IT	avg.	DE	EN	FR	IT	avg.
1	**bert-base-multilingual (job ads, w/o context)**	0.9218	0.9244	0.9214	0.9225	0.9132	**0.9420**	**0.9517**	**0.9311**	**0.9345**
2	bert-base-multilingual (sentences, w/ context)	0.9176	0.9243	0.9218	0.9212	0.9118	0.8909	0.9220	0.9152	0.9100
3	bert-base-multilingual (job ads, 2-phase)	0.9229	0.9255	0.9235	0.9240	0.9129	0.9401	0.9500	0.9257	0.9322
4	bert-jobad (job ads, w/o context, 2-phase)	0.9111	0.9054	0.9049	0.9071	0.9194	0.8962	0.8837	0.8717	0.8928
5	xlm-roberta-base (sentences, w/ context)	0.9235	**0.9308**	0.9291	0.9278	0.9170	0.9226	0.9319	0.9080	0.9199
6	xlm-roberta-base (sentences, w/o context)	0.9178	0.9236	0.9225	0.9213	0.9089	0.9254	0.9180	0.9034	0.9139
7	xlm-roberta-base (sentences, w/ context, 2-phase)	0.9235	**0.9308**	**0.9300**	0.9281	0.9154	0.9151	0.9303	0.9014	0.9156
8	xlm-roberta-base (job ads, w/o context)	**0.9251**	0.9292	0.9291	0.9278	0.9177	0.9294	0.9331	0.9134	0.9234
9	**xlm-roberta-jobad (job ads, w/o context, finetune)**	0.9243	0.9307	0.9295	**0.9282**	**0.9202**	0.9323	0.9295	0.9211	0.9258
	average	0.9208	0.9250	0.9235	0.9217	0.9152	0.9216	0.9278	0.9101	0.9170

Table 7: Table of Model Performance

The reason behind could be the fact that the whole job advertisement contains more context information and is more coherent, which helps the models produce better results. As a matter of fact, the discussion and analysis of model performance were based on the accuracy scores which were evaluated on the test set with the whole job advertisement.

Table 7 presents the models performance evaluated on the silver and gold test set based on the whole job advertisements. The silver test sets stem from the silver standard data, while the gold test test contains the manually corrected predictions and the original German gold standard. The differences of size of silver test set in each language are minor. However, despite being categorized together with the German test set as the gold test set, the amount of testing samples in English, French and Italian is noteworthy lower. Models generated by 2-phase fine-tune process are also grouped together for a better view, which is to say, model 3, 4 and 7 are actually 4 models fine-tunes by monolingual data, and the accuracy score is reported on the corresponding test set in the same language. The scores in **bold** indicate the highest score in each row (per language).

Overall, the `bert-base-mutilingual` based Model 1, which was trained on the whole job advertisements, achieved the best accuracy scores on the gold test set with an average of 0.9345, leaving a margin by 0.04 compared to the worst model 4. Also considering the competitive average accuracy on the silver test set, this implies the versatility and robustness of the `bert-base-multilingual` language model. However, on account of the origin of the gold test set, which is prediction from the model 1, it is hard not to suspect that the gold test set has a bias towards model 1. Nevertheless, model 9, which was based on the `xlm-roberta-base` with pre training via in-domain data, accomplished the best accuracy score on the silver test set by the average of 0.9282. Model 9 also has the best accuracy score on the gold test set, if `bert-base-multilingual` based models were omitted due to potential bias.

Comparing model 1 vs model 2, as well as model 5 vs model 8 should answer the research question 2. Models based on the same word embedding but on different types of training data have different performance, and the results suggest that training on whole job advertisements brings benefits to the models. In terms of research question 3, XLM-RoBERTa is reported to have 2-20% improvement over BERT, and the results support this claim. Taking only silver standard data into consideration, models based on `xlm-roberta-base` (5-8) have a tendency of higher accuracy scores than models based on `bert-base-multilingual-cased` (1-3). Research question 4 regards the effectiveness of 2-phase training in this multilingual set up. The results could not come to an agreement. Models of 2-phase training tend to have a better performance on the silver test set (model 3 vs model 1, model 7 vs model 5), while a worse performance on the gold test set. This suggests that the validity of 2-phase training is limited in the experiments.

5.2.3 Further Analysis

Since table 7 shows that the differences of accuracy score between each model are not substantial by any means, it is necessary to dive into the predictions of the models to get a better understanding of model performance. Due to the fact that the silver test set was synthesized, and the gold test set in English, French and Italian is short in size, the German gold test set serves the purpose to perform a fine-grained analysis. Figure 13 shows the confusion matrix of the predictions from model 9 with highest accuracy score on a test set in German, which is based on the `xlm-roberta-base` with in-domain pretraining. Following the same set up as the plots for word aligners,

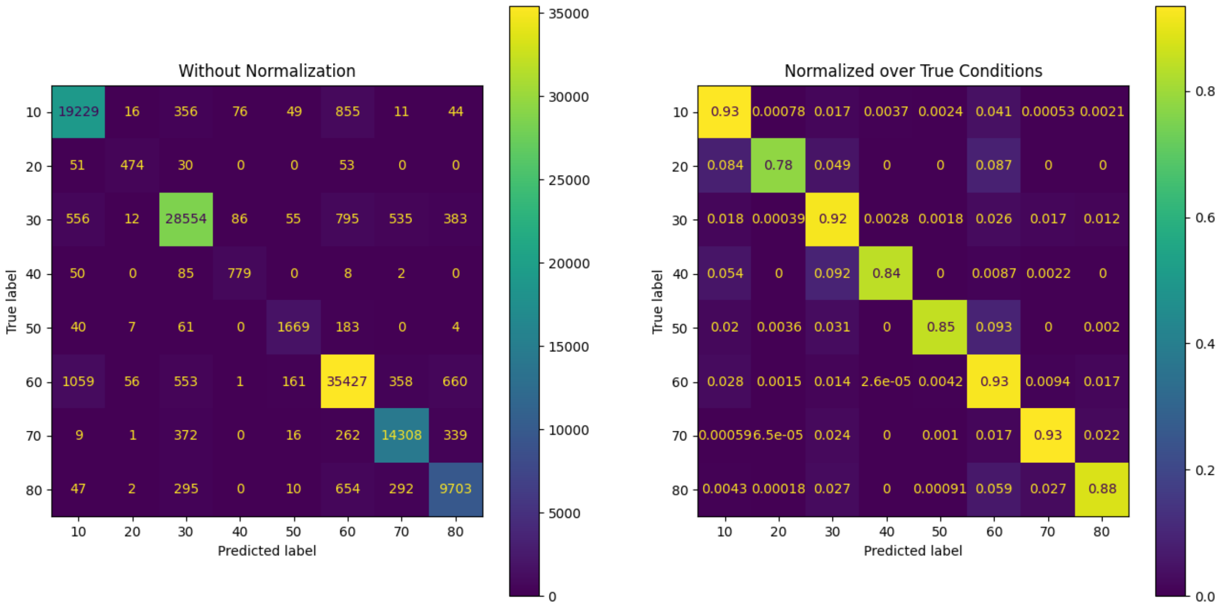

Figure 13: Confusion Matrix of DE_gold of `xlm-roberta-base-job`

the subplot on the left shows the values without normalization (absolute numbers), and the right subplot shows the values normalized over the true condition (values add up to 1 in each row). In additonal to this, Figure 14 illustrates the label distribution in the test set in German. Figure 13 indicates that despite the label imbalance of data, the model achieved relatively good recall for each text zone. Zone 20 (reason of vacancy) has the worst average recall score of 0.78, however, zone 20 composes the least of the text zones, as shown in the figure below. The confusion matrix also demonstrates a correlation between the number of presence and the average recall score. Zone 20, 40, 50, which compose a small number of samples in the test set, all have recall scores below 0.8. On the other hand, zones 10, 30, 60 have the average recall above 0.9, and they are the zones with the dominantly more samples in the test set. Additionally, the detailed recall for each text zone and plots of confusion matrices are included in appendix C.

- Case 1

  true `<70>`5 + years in a management consultancy or in a strategic planning department of a multinational .`</70>`
  pred `<60>`5 + years in a management consultancy or in a strategic planning department of a multinational .`</60>`

- Case 2

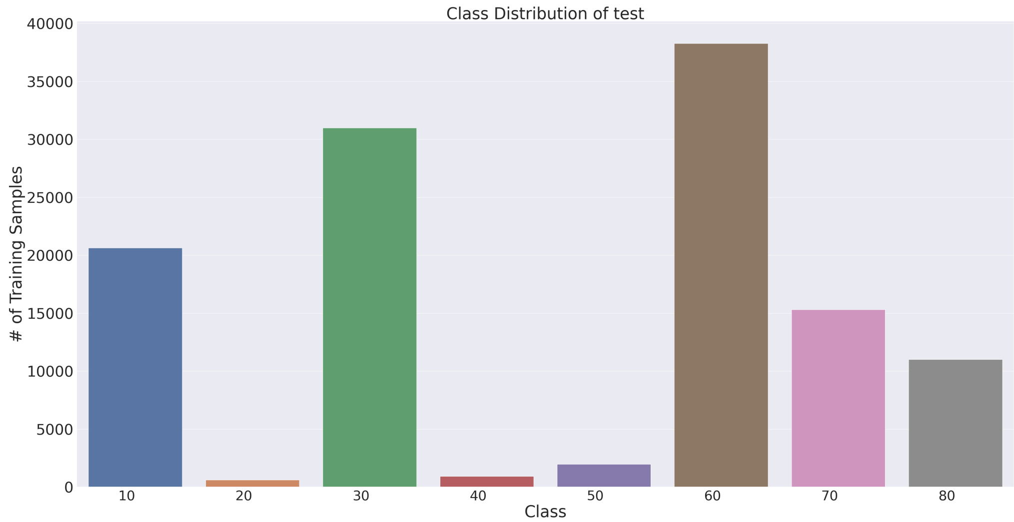

Figure 14: Label Distribution in the German Test Set

true <30>Wir bieten Ihnen</30><10>die Möglichkeit , in unseren 18 Agenturen sowie an unserem Hauptsitz in Luzern , Ihre Kompetenzen und Ideen einzubringen .</10>
pred <30>Wir bieten Ihnen die Möglichkeit ,</30><10>in unseren 18 Agenturen sowie an unserem Hauptsitz in Luzern ,</10><70>Ihre Kompetenzen</70><60>und Ideen einzubringen .</60>

- Case 3

true <10>With a passion to understand consumers' preferences and a relentless drive to innovate , Givaudan is at the forefront of creating flavours and fragrances that ' engage your senses ' .</10>
pred <80>With a passion to</80><60>understand consumers' preferences</60><10>and a relentless drive to innovate , Givaudan is at the forefront of creating flavours and fragrances that ' engage your senses ' .</10>

Some further error analysis was also carried out to have a better understanding of the model predictions. The 3 cases above are selected samples from the predictions of the gold test set based on sentences in English generated by the model 1. Each sample contains 1 sentence of the gold test set (true) and predictions (pred) with XML tags injected for a better view. In case 1, the model has the ability to predict

		Silver Test Set				Gold Test Set				
No.	Model Description	EN	FR	IT	avg.	DE	EN	FR	IT	avg.
4	bert-jobad (job ads, w/o context, 2-phase)	0.9111	0.9054	0.9049	0.9071	0.9194	0.8962	0.8837	0.8717	0.8928
10	bert-jobad (original, w/o finetune)	0.6203	0.4144	0.4512	0.4953	0.9152	0.7055	0.4143	0.4660	0.6253

Table 8: Model Performance of Original `bert-jobad`

all tokens in the sentences with the same text zone labels, showing no issues of segmentation. Tokens were marked as job description (zone 60) instead of required hard skills (zone 70). It suggests that the model can not identify the intention of the sentences. Case 2 shows a sentence with multiple text zone labels. Zone 30 stands for administration and residual text and zone 10 stands for company description. The model had different segmentations regarding zone 30 and 10, also it added zone 70 and 60. The latter seems to be caused by the capitalized German "Ihre", making the model treating the second part of the sentence as a new and perform the predictions upon.

Case 3 presents an interesting example as well. Reading the whole sentence, it is effortless for humans to identify the subject of the sentence is the company "Givaudan", and the rest of text supports the company description. The model may have had a hard time identifying the essential subject of the sentence and added zone 80 (required personality, soft skills) and 60, showing the lack of capability to catch the built-in connections within sentences. The generalizability of the results is limited by the source of the test set, which is sentence-based. As mentioned in the previous section, models tend to have better performance on job-advertisements based test sets, where the latter could provide more valuable context information. As a matter of fact, case 1 was indeed predicted with correct zone labels in the job-advertisements based test set. However, inspecting zone errors in whole job advertisements is cumbersome due to the large size of text, and sentence-based analysis are much more intuitive. Further research is needed to establish better error analysis strategies.

Table 8 shows the model 10's performance, which is the original zone tagger of model 4. Domain-adapted monolingual language model `bert-base-german-cased` is the foundation of Model 10 and model 4 performed a 2-phase fine-tune process on monolingual data in English, French and Italian. Comparing two models accuracy scores on test set in non-German languages, it implies that model 4 gained ability to conduct sequence labeling on English, French and Italian, even so the original model is only trained on German data. The results might suggest that even monolingual Transformer languages models have the capability to learn generalized presentations across different languages. It is yet beyond the scope of this study to dive into the extensibility of Transformer language models and avenues for future research could include this aspect.

5.3 Overall Evaluation and Discussion

Considering the outcomes of the word alignment and zone tagging experiments, the inquiry posed by research question 1 can be answered. The results of the accuracy scores suggest that the word aligners possess better performance than the zone taggers. The accuracy of fast-align on the silver test set was found to be 0.97, representing an improvement of 5% over the highest score of 0.92 obtained by zone taggers (model 9). However, as illustrated in section 5.1, the reliability of the performance from word aligners is limited and requires further assessment. Given the absence of accuracy scores on the gold test set, it is difficult to meaningfully compare the performance of a word alignment approach and a zone tagger approach. Moreover, the trained zone taggers can conduct predictions directly, while word aligners have to rely on the parallel translation with text zone labels from source language. The practical application of word aligners is restricted.

In general, trained models exhibit satisfactory performance on both silver and gold test sets. Most zone taggers have been demonstrated to offer accuracy scores that are higher than 0.91, and have proven to have some capacity to address the disparity in labeling of data. In terms of training process, the time and effort consumption was also acceptable. Initial training typically demanded 50 hours of labor, whereas the fine-tuning process necessitated significantly less time. This implies that the methodology based on model training is suitable for a production environment, and can be advantageous for the multinational area labeling system. The performance distinctions among the nine trained models were not notably disparate. Model 1 and 9 achieved the highest scores on the gold and silver test sets, respectively, though the difference between them and the other models was marginal (1% - 4%). This experiment demonstrates that the selection of machine learning models may not significantly influence the accuracy of text zone labeling at this stage. A significant improvement may be realized through other considerations such as the quality of the labeled data.

6 Conclusion

The presented research aimed to test approaches for cross-lingual projection of text zoning labels from German job advertisements to other languages. For the conduction of experiments, i.e., word alignment and model training, a silver standard dataset was created via the state-of-the-art machine translation engine DeepL. By conducting several experiments using approaches empowered by word alignment and sequence labeling model training based on the silver standard data, this work tried to answer the central questions for the research as follows:

1. In terms of the accuracy of text zoning label projection, to what extent do the performances of the word alignment approach and the zone tagger approach differ from each other?

   The performance of the word alignment and the zone tagger approaches are barely comparable based on the experiments. The evaluation metrics cannot be leveled due to the lack of a proper gold test set. On the other hand, the higher accuracy score on the silver test set from zone taggers shows the better usability when the parallel translation data is presented. But this cannot certify the correlation of higher quality compared to zone taggers in any other circumstance.

2. For multilingual zone taggers, will the different segmentation of training material play a role here, i.e., training on the unit of whole job advertisements or sentences?

   For zone taggers, the different segmentation of training material play a role in the experiments. Models based on the same word embedding but on different types of training data have various performances. The results suggest that training on full job advertisements brings benefits to the models. A possible reason could be that the valuable context information is preserved, furthermore, linear structure of zones in a job ad are often similar. Header and footers of job ads, for instance.

3. Which foundational models are better, i.e., can the superiority of word em-

beddings from different language models be observed (BERT versus XLM-RoBERTa)?

The superiority of word embeddings from different language models, i.e., BERT versus XLM-RoBERTa, can be observed in this case. XLM-RoBERTa-based models tend to have better performance than BERT based. But the level of distinctness is minimal (less than 5% in terms of accuracy score) and is only noticeable on the silver standard data, on the other hand, the small size of the gold standard test set is too limited to draw final conclusions.

4. Will 2-phase training, i.e., to fine-tune multilingual zone taggers with monolingual data, will deliver improved results??

   The enhancement of 2-phase training for multilingual zone taggers in the monolingual scenario is not clearly noticeable, since the results show contradictory results from 2 different model groups. The only improvements can be observed from the fine-tuning process of zone taggers originally trained on monolingual data, but this is beyond the scope of this work and needs further research.

5. What are the particular characteristics of the model predictions on the test set?

   Error analysis has revealed that the model's inability to detect semantic connections between sentences is a notable limitation of its performance on the test data set. This resulted in some incorrect attributions and misclassifications of zone labels. The segmentation issues are not indicative of the comparative results; however, the text zone labels with minimal representation lead to segmentation mistakes, which is exacerbated by the data imbalance issue.

This research provides new insight into mitigating the problem raised by labeled data acquisition bottleneck, focusing on the cross-lingual projection of text zoning labels. The creation process of silver standard data clearly illustrates the effectiveness of the translation engine DeepL. With the minimal cost of time and funding compared to human resources, the translation engine produces sufficient data with good quality that built the base for the word alignment and model training process in the following experiments. While the proper evaluation data limits the generalizability of the results produced by the word aligners, this approach provides a new understanding of the ability of the methods based on word alignment. On the other hand, trained zone taggers show the versatility and robustness of the machine learning pipeline regarding sequence labeling as well as the Transformer-based language models. Furthermore, the implementation of the training data type comparison, 2-phase training, and fine-tuning on domain-adapted word embeddings shed light on the impacts of different machine learning techniques for studying computational linguistics.

This work has several contributions to the study of cross-lingual transfer. First, it generated a multilingual corpus via the machine translation service, which can be utilized for further model training, domain adaption, and evaluation. Second, nine sequence labeling models were trained in the experiments with competitive performance, which can be further studied and implemented for the text zoning tasks for SJMM. In addition, the research addressed the knowledge gap of the practicality of the machine learning models trained on the silver standard data. Finally, the research findings provided valuable insight into the methods regarding the model training on synthesized data to address the labeled data acquisition bottleneck.

Future Work Future studies could focus on a better preprocess of the original data in column format to better understand the implications of these results. The raw form of the initially collected job advertisements does not contain sentence-separating information, making it difficult to directly adopt the data into many mainstream NLP tools since most tools are developed based on sentences. Especially in terms of encoding sentences with Transformer embeddings, which usually have a length of 768 or more, a long job advertisement usually cannot easily fit into the GPU's memory. On the other hand, many collected job advertisements are organized in bullet points fashion or lack explicit sentence separators like full stops or question marks. For this work, a rule-based pipeline was developed to cooperate with the XML tag injection, which delivered, in most cases, satisfactory results yet still with some flaws, e.g., too short sentences when splitting bullet points or too long sentences when the job advertisements do not have full stops. The text zoning task could definitely benefit from a better preprocessing of the raw job advertisement data in the future. Additionally, the efficacy of zone taggers is diminished when transitioning from silver standard test set assessment to original test set assessment, which may be attributable to the limited size of the gold standard test set. Conducting a proper gold test could increase confidence in the evaluation and provide a more holistic overview of the models' performance.

References

F. Achard, G. Vaysseix, and E. Barillot. XML, bioinformatics and data integration . *Bioinformatics*, 17(2):115–125, 02 2001. ISSN 1367-4803. doi: 10.1093/bioinformatics/17.2.115. URL `https://doi.org/10.1093/bioinformatics/17.2.115`.

A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, 2019.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible markup language (xml) 1.0 (fifth edition). W3C Recommendation, 2008. Available at `http://www.w3.org/TR/REC-xml/`.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

A. Conneau, R. Rinott, G. Lample, A. Williams, S. R. Bowman, H. Schwenk, and V. Stoyanov. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL `https://aclanthology.org/2020.acl-main.747`.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Z.-Y. Dou and G. Neubig. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online, Apr. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.181. URL `https://aclanthology.org/2021.eacl-main.181`.

C. Dyer, V. Chahuneau, and N. A. Smith. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL `https://aclanthology.org/N13-1073`.

M. Ehrmann, M. Turchi, and R. Steinberger. Building a multilingual named entity-annotated corpus using annotation projection. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 118–124, Hissar, Bulgaria, Sept. 2011. Association for Computational Linguistics. URL `https://aclanthology.org/R11-1017`.

A. Galassi, K. Drazewski, M. Lippi, and P. Torroni. Cross-lingual annotation projection in legal texts. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 915–926, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.79. URL `https://aclanthology.org/2020.coling-main.79`.

A.-S. Gnehm. Text zoning for job advertisements with bidirectional LSTMs. In *Gnehm, Ann-Sophie (2018). Text zoning for job advertisements with bidirectional LSTMs. In: 3rd Swiss Text Analytics Conference - SwissText 2018, Winterthur, 12 June 2018 - 13 June 2018, 1-9.*, pages 1–9, Winterthur, June 2018. University of Zurich. doi: 10.5167/uzh-186646. URL `http://ceur-ws.org/Vol-2226/`.

A.-S. Gnehm and S. Clematide. Text Zoning and Classification for Job Advertisements in German, French and English. In *Proceedings of the Fourth Workshop on Natural Language Processing and Computational Social Science*, pages 83–93, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlpcss-1.10. URL `https://aclanthology.org/2020.nlpcss-1.10`.

A.-S. Gnehm, E. Bühlmann, and S. Clematide. Evaluation of transfer learning and domain adaptation for analyzing German-speaking job advertisements. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3892–3901, Marseille, France, June 2022. European Language Resources Association. URL `https://aclanthology.org/2022.lrec-1.414`.

X. Han and J. Eisenstein. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1433. URL `https://aclanthology.org/D19-1433`.

G. Hanneman and G. Dinu. How should markup tags be translated? In *Proceedings of the Fifth Conference on Machine Translation*, pages 1160–1173, Online, Nov. 2020. Association for Computational Linguistics. URL `https://aclanthology.org/2020.wmt-1.138`.

K. Hashimoto, R. Buschiazzo, J. Bradbury, T. Marshall, R. Socher, and C. Xiong. A high-quality multilingual dataset for structured documentation translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 116–127, Florence, Italy, Aug. 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5212. URL `https://aclanthology.org/W19-5212`.

X. Li, G. Li, L. Liu, M. Meng, and S. Shi. On the word alignment from neural machine translation. In *Proceedings of the 57th Annual Meeting of the*

Association for Computational Linguistics, pages 1293–1303, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1124. URL `https://aclanthology.org/P19-1124`.

B. Marie and A. Fujita. Synthesizing monolingual data for neural machine translation. *ArXiv*, abs/2101.12462, 2021.

M. Müller. Treatment of markup in statistical machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 36–46, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4804. URL `https://aclanthology.org/W17-4804`.

S. Schweter and A. Akbik. Flert: Document-level features for named entity recognition. *ArXiv*, abs/2011.06993, 2020.

H. Sluyter-Gäthje, P. Bourgonje, and M. Stede. Shallow discourse parsing for under-resourced languages: Combining machine translation and annotation projection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1044–1050, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL `https://aclanthology.org/2020.lrec-1.131`.

Z. Tan, S. Wang, Z. Yang, G. Chen, X. Huang, M. Sun, and Y. Liu. Neural machine translation: A review of methods, resources, and tools. *AI Open*, 1: 5–21, 2020. ISSN 2666-6510. doi: https://doi.org/10.1016/j.aiopen.2020.11.001. URL `https://www.sciencedirect.com/science/article/pii/S2666651020300024`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

S. Weber and M. Steedman. Zero-shot cross-lingual transfer is a hard baseline to beat in German fine-grained entity typing. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 42–48, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.insights-1.7. URL `https://aclanthology.org/2021.insights-1.7`.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. 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, Oct. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL `https://aclanthology.org/2020.emnlp-demos.6`.

D. Yarowsky and G. Ngai. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*, 2001. URL `https://aclanthology.org/N01-1026`.

T. Zenkel, J. Wuebker, and J. DeNero. Automatic bilingual markup transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3524–3533, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.299. URL `https://aclanthology.org/2021.findings-emnlp.299`.

A Experiment Results of Word Aligners

This appendix contains the detailed results and confusion matrices of 3 word aligners.

- English Test Set

Aligner	Precision (micro)	Recall (macro)	Accuracy	F1 (micro)
`fast_align`	0.980730	0.971031	0.980730	0.980730
`awesome-align`	0.990205	0.987415	0.990205	0.990205
`awesome-align` (fine-tuned)	0.992860	0.991032	0.992860	0.992860

- French Test Set

Aligner	Precision (micro)	Recall (macro)	Accuracy	F1 (micro)
`fast_align`	0.969938	0.954592	0.969938	0.969938
`awesome-align`	0.982563	0.975066	0.982563	0.982563
`awesome-align` (fine-tuned)	0.981853	0.974853	0.981853	0.981853

- Italian Test Set

Aligner	Precision (micro)	Recall (macro)	Accuracy	F1 (micro)
`fast_align`	0.966008	0.949758	0.966008	0.966008
`awesome-align`	0.979998	0.972885	0.979998	0.979998
`awesome-align` (fine-tuned)	0.977965	0.973474	0.977965	0.977965

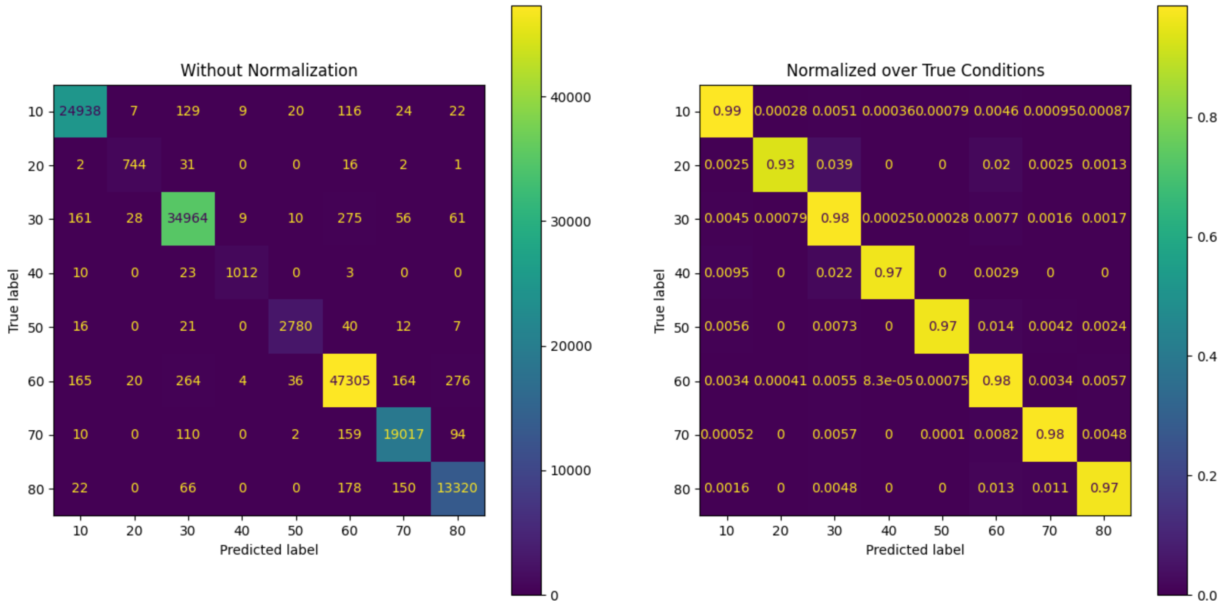

Figure 15: Confusion Matrix of Predictions of `fast_align` in English

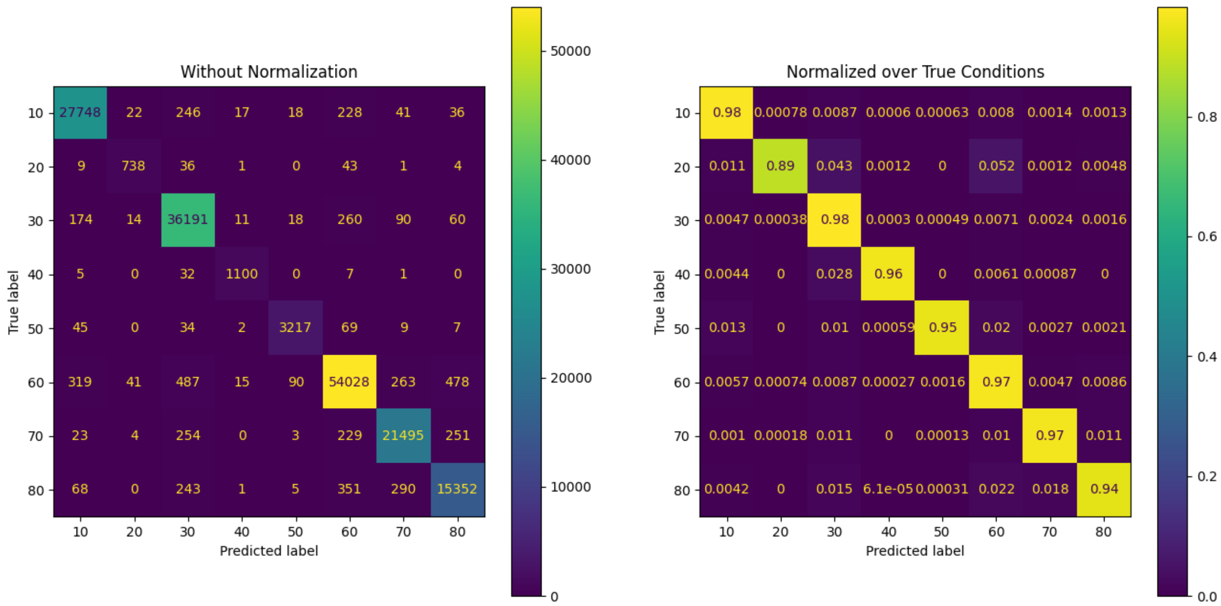

Figure 16: Confusion Matrix of Predictions of original `fast_align` in French

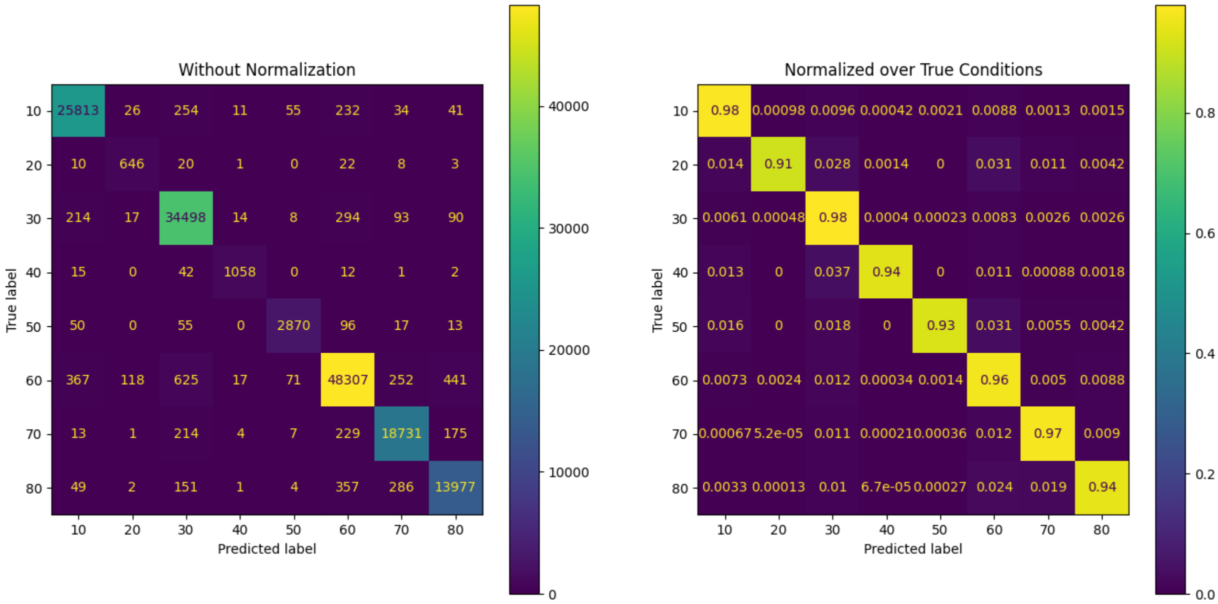

Figure 17: Confusion Matrix of Predictions of original `fast_align` in Italian

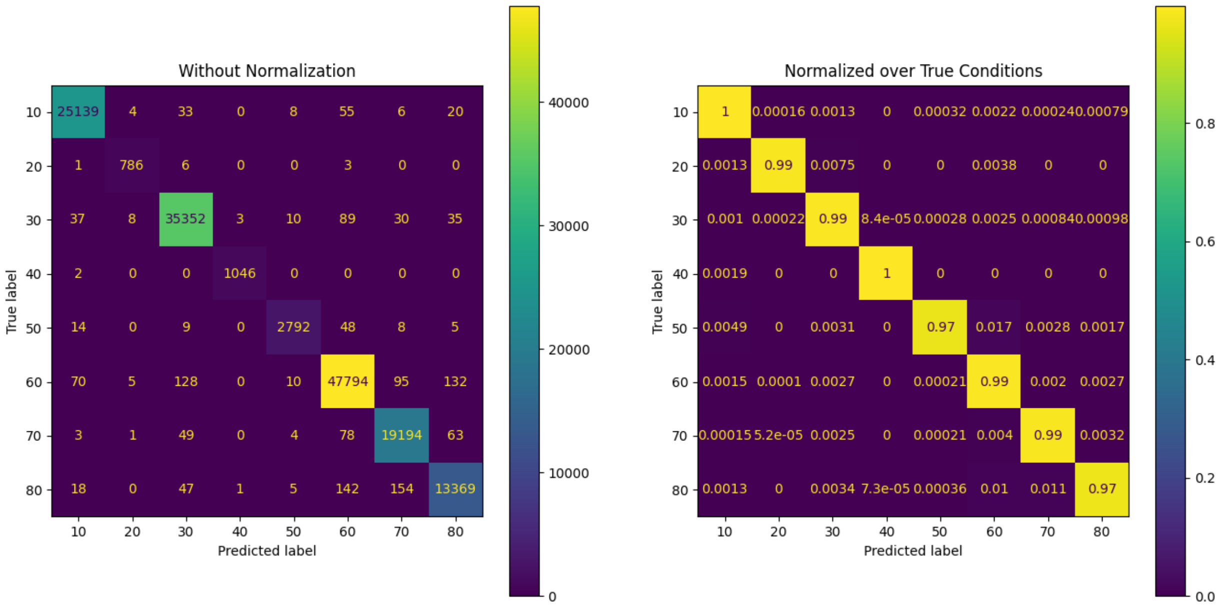

Figure 18: Confusion Matrix of Predictions of original `awesome-align` in English

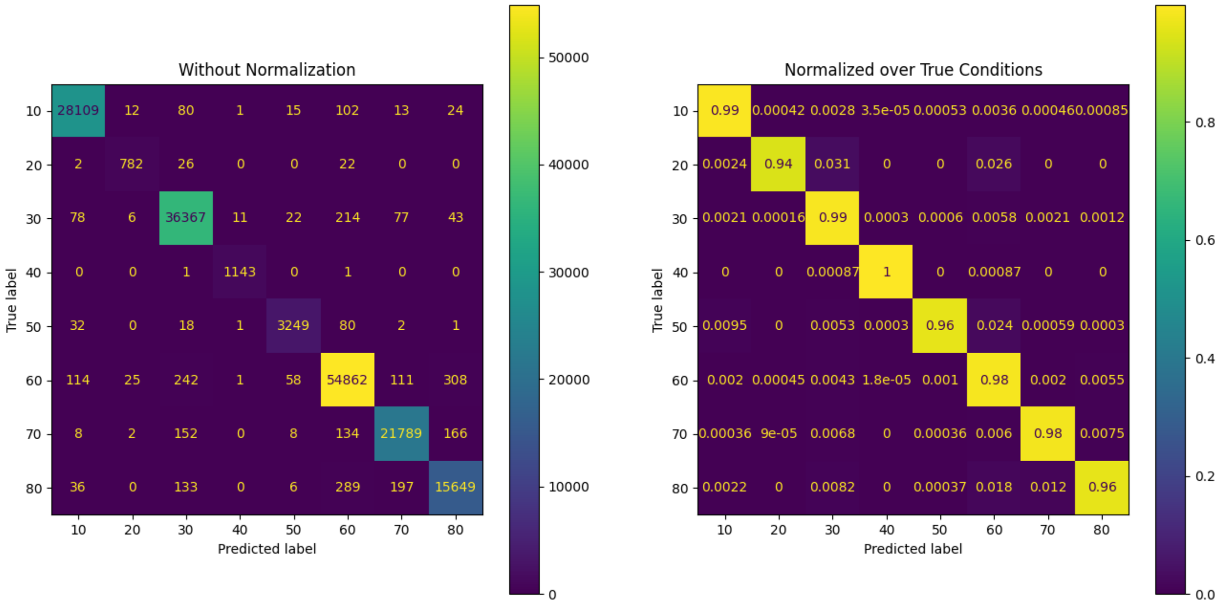

Figure 19: Confusion Matrix of Predictions of original `awesome-align` in French

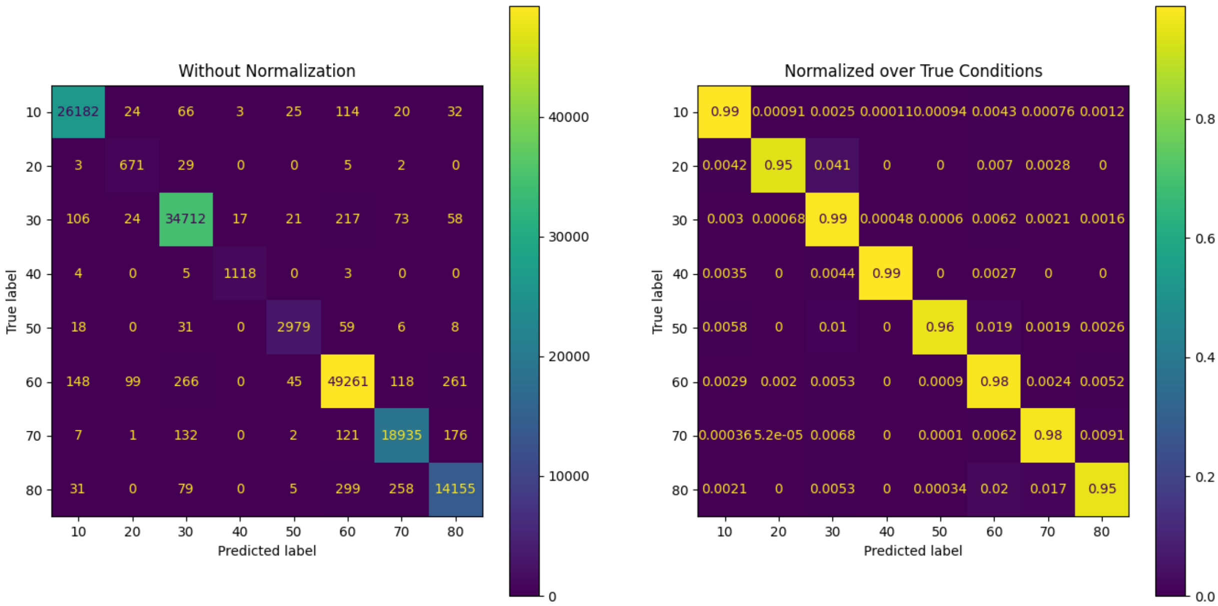

Figure 20: Confusion Matrix of Predictions of original `awesome-align` in Italian

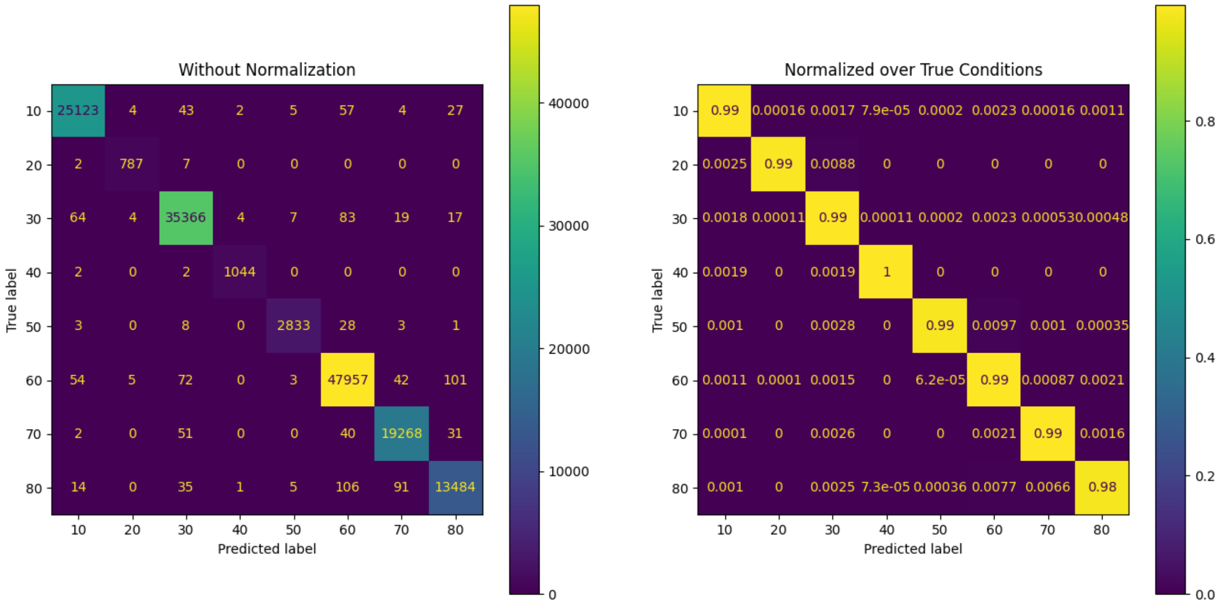

Figure 21: Confusion Matrix of Predictions of fine-tuned `awesome-align` in English

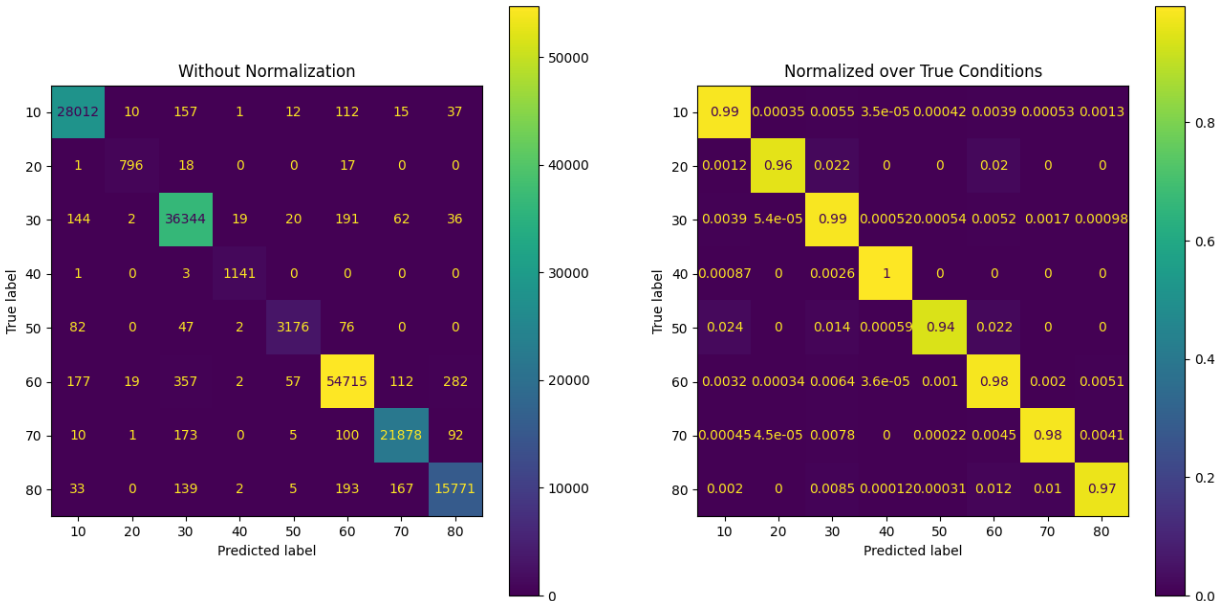

Figure 22: Confusion Matrix of Predictions of fine-tuned `awesome-align` in French

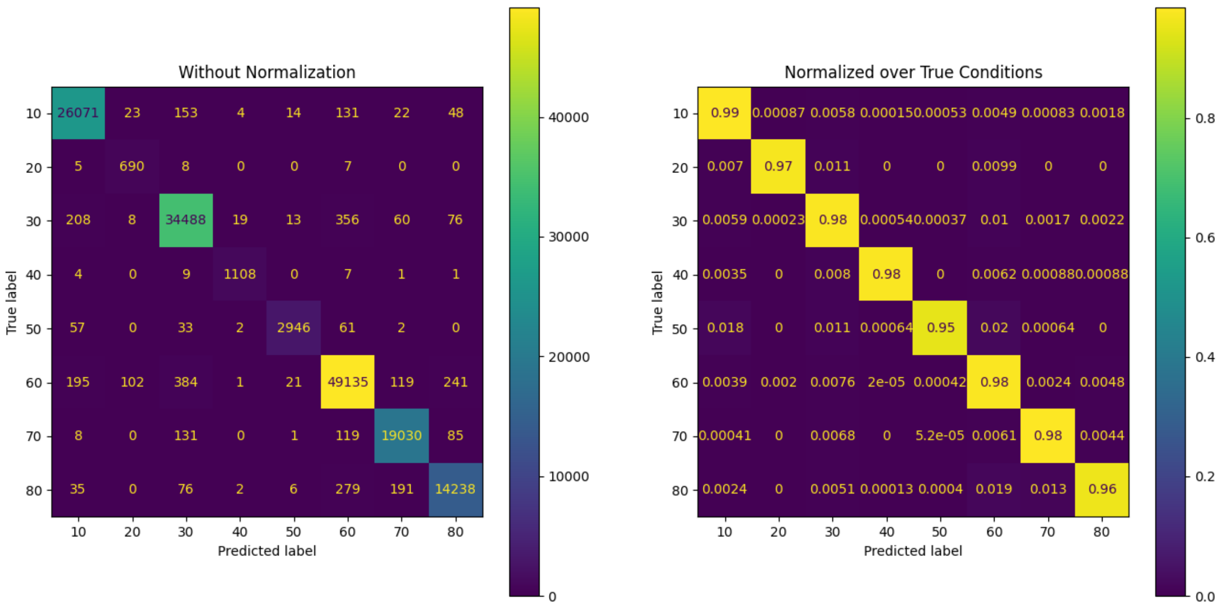

Figure 23: Confusion Matrix of Predictions of fine-tuned `awesome-align` in Italian

B Detailed Statistics of Model Training

This appendix contains the detailed statistics, such as tables and plots of training loss.

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.579890	0.343805	0.8878	0.8878	0.8878	0.8878
2	0.417876	0.290814	0.9073	0.9073	0.9073	0.9073
3	0.373582	0.281022	0.9122	0.9122	0.9122	0.9122
4	0.346783	0.270932	0.9150	0.9150	0.9150	0.9150
5	0.326311	0.274587	0.9162	0.9162	0.9162	0.9162
6	0.310105	0.276682	0.9173	0.9173	0.9173	0.9173
7	0.296396	0.272922	0.9197	0.9197	0.9197	0.9197
8	0.284912	0.274733	0.9196	0.9196	0.9196	0.9196
9	0.274967	0.281857	0.9207	0.9207	0.9207	0.9207
10	0.266809	0.284385	0.9205	0.9205	0.9205	0.9205
11	0.259306	0.287663	0.9208	0.9208	0.9208	0.9208
12	0.253438	0.290884	0.9206	0.9206	0.9206	0.9206
13	0.248554	0.297159	0.9209	0.9209	0.9209	0.9209
14	0.244680	0.298648	0.9212	0.9212	0.9212	0.9212
15	0.241440	0.301746	0.9215	0.9215	0.9215	0.9215
16	0.239201	0.304747	0.9211	0.9211	0.9211	0.9211
17	0.237516	0.305664	0.9214	0.9214	0.9214	0.9214
18	0.236440	0.305369	0.9215	0.9215	0.9215	0.9215
19	0.235559	0.306476	0.9215	0.9215	0.9215	0.9215
20	0.235364	0.306373	0.9215	0.9215	0.9215	0.9215

Table 9: Training Statistics of `bert-base-multilingual-cased`

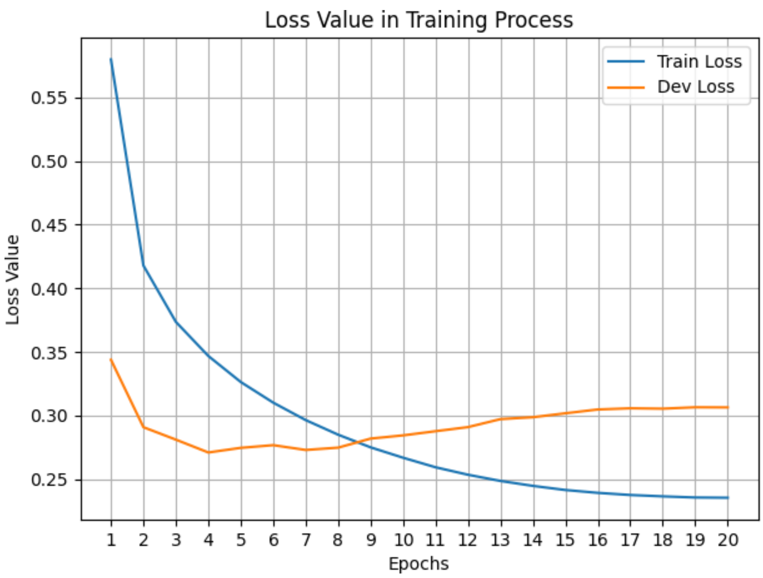

Figure 24: Training Loss of `bert-base-multilingual-cased`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.459916	0.322458	0.8991	0.8991	0.8991	0.8991
2	0.366618	0.283818	0.9137	0.9137	0.9137	0.9137
3	0.320049	0.287561	0.9167	0.9167	0.9167	0.9167
4	0.290154	0.299754	0.9173	0.9173	0.9173	0.9173
5	0.266763	0.317601	0.9180	0.9180	0.9180	0.9180
6	0.248796	0.325867	0.9182	0.9182	0.9182	0.9182
7	0.235054	0.340508	0.9187	0.9187	0.9187	0.9187
8	0.225276	0.347198	0.9190	0.9190	0.9190	0.9190
9	0.219524	0.351007	0.9192	0.9192	0.9192	0.9192
10	0.216745	0.352952	0.9191	0.9191	0.9191	0.9191

Table 10: Training Statistics of `bert-base-multilingual-cased_w_context`

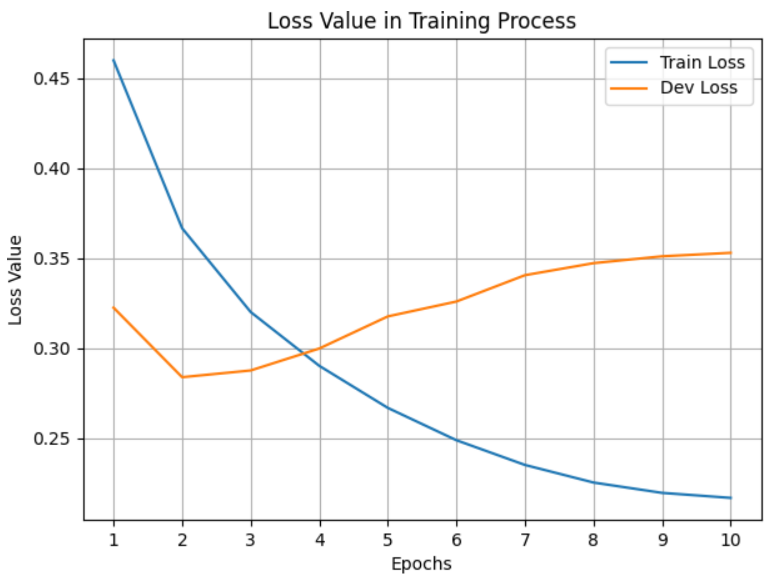

Figure 25: Training Loss of `bert-base-multilingual-cased_w_context`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.268481	0.322540	0.9089	0.9089	0.9089	0.9089
2	0.263612	0.323366	0.9117	0.9117	0.9117	0.9117
3	0.256620	0.319890	0.9126	0.9126	0.9126	0.9126
4	0.250496	0.325896	0.9107	0.9107	0.9107	0.9107
5	0.245339	0.325698	0.9120	0.9120	0.9120	0.9120
6	0.240652	0.332302	0.9128	0.9128	0.9128	0.9128
7	0.236369	0.335088	0.9117	0.9117	0.9117	0.9117
8	0.233779	0.338095	0.9120	0.9120	0.9120	0.9120
9	0.230854	0.344266	0.9120	0.9120	0.9120	0.9120
10	0.229324	0.341967	0.9121	0.9121	0.9121	0.9121

Table 11: Training Statistics of `bert-base-multilingual-cased_2_DE`

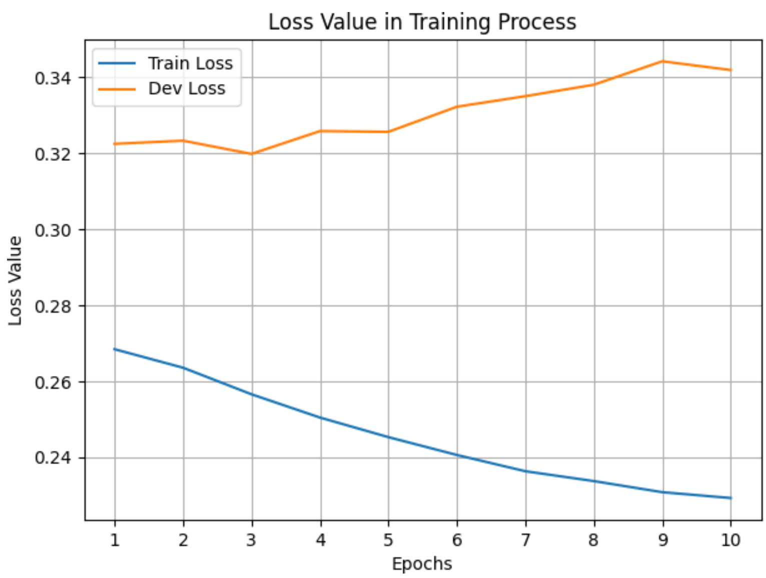

Figure 26: Training Loss of `bert-base-multilingual-cased_2_DE`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.247232	0.301375	0.9192	0.9192	0.9192	0.9192
2	0.244370	0.296231	0.9205	0.9205	0.9205	0.9205
3	0.237111	0.303246	0.9204	0.9204	0.9204	0.9204
4	0.230649	0.309163	0.9217	0.9217	0.9217	0.9217
5	0.225328	0.316513	0.9212	0.9212	0.9212	0.9212
6	0.221057	0.317100	0.9216	0.9216	0.9216	0.9216
7	0.216929	0.315226	0.9226	0.9226	0.9226	0.9226
8	0.214098	0.322788	0.9228	0.9228	0.9228	0.9228
9	0.211294	0.320047	0.9223	0.9223	0.9223	0.9223
10	0.209802	0.321899	0.9222	0.9222	0.9222	0.9222

Table 12: Training Statistics of `bert-base-multilingual-cased_2_EN-US`

Figure 27: Training Loss of `bert-base-multilingual-cased_2_EN-US`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.232948	0.300724	0.9246	0.9246	0.9246	0.9246
2	0.229316	0.304373	0.9251	0.9251	0.9251	0.9251
3	0.223716	0.299805	0.9264	0.9264	0.9264	0.9264
4	0.217222	0.310755	0.9263	0.9263	0.9263	0.9263
5	0.212037	0.313568	0.9261	0.9261	0.9261	0.9261
6	0.208016	0.319212	0.9260	0.9260	0.9260	0.9260
7	0.204619	0.324803	0.9266	0.9266	0.9266	0.9266
8	0.200590	0.321582	0.9268	0.9268	0.9268	0.9268
9	0.198761	0.328388	0.9267	0.9267	0.9267	0.9267
10	0.196976	0.328474	0.9265	0.9265	0.9265	0.9265

Table 13: Training Statistics of `bert-base-multilingual-cased_2_FR`

Figure 28: Training Loss of `bert-base-multilingual-cased_2_FR`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.238958	0.288385	0.9244	0.9244	0.9244	0.9244
2	0.236108	0.290240	0.9236	0.9236	0.9236	0.9236
3	0.228774	0.297473	0.9240	0.9240	0.9240	0.9240
4	0.222315	0.301879	0.9251	0.9251	0.9251	0.9251
5	0.216706	0.300945	0.9252	0.9252	0.9252	0.9252
6	0.212791	0.307052	0.9247	0.9247	0.9247	0.9247
7	0.208393	0.312129	0.9251	0.9251	0.9251	0.9251
8	0.204543	0.311386	0.9253	0.9253	0.9253	0.9253
9	0.202361	0.313855	0.9252	0.9252	0.9252	0.9252
10	0.201422	0.315846	0.9255	0.9255	0.9255	0.9255

Table 14: Training Statistics of `bert-base-multilingual-cased_2_IT`

Figure 29: Training Loss of `bert-base-multilingual-cased_2_IT`

EPOCH	TRAIN_LOSS	DEV_LOSS	DEV_PRECISION	DEV_RECALL	DEV_F1	DEV_ACCURACY
1	0.769109	0.487757	0.8484	0.8484	0.8484	0.8484
2	0.506219	0.339182	0.8932	0.8932	0.8932	0.8932
3	0.434156	0.322104	0.9003	0.9003	0.9003	0.9003
4	0.402715	0.314972	0.9029	0.9029	0.9029	0.9029
5	0.382593	0.308046	0.9053	0.9053	0.9053	0.9053
6	0.367776	0.308234	0.9057	0.9057	0.9057	0.9057
7	0.356873	0.308582	0.9075	0.9075	0.9075	0.9075
8	0.349634	0.307753	0.9079	0.9079	0.9079	0.9079
9	0.342976	0.308946	0.9084	0.9084	0.9084	0.9084
10	0.339593	0.309014	0.9086	0.9086	0.9086	0.9086

Table 15: Training Statistics of `jobad_bert_finetune_multi`

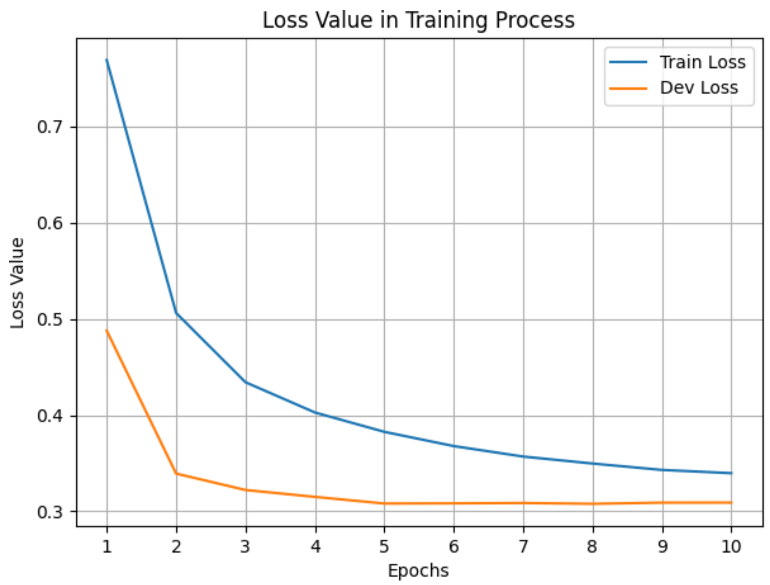

Figure 30: Training Loss of `jobad_bert_finetune_multi`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.626615	0.333337	0.8984	0.8984	0.8984	0.8984
2	0.372538	0.281417	0.9149	0.9149	0.9149	0.9149
3	0.323042	0.280869	0.9196	0.9196	0.9196	0.9196
4	0.294256	0.282612	0.9215	0.9215	0.9215	0.9215
5	0.272238	0.295084	0.9210	0.9210	0.9210	0.9210
6	0.254922	0.300806	0.9233	0.9233	0.9233	0.9233
7	0.241477	0.313945	0.9234	0.9234	0.9234	0.9234
8	0.230648	0.324930	0.9231	0.9231	0.9231	0.9231
9	0.222960	0.329864	0.9230	0.9230	0.9230	0.9230
10	0.217252	0.334335	0.9233	0.9233	0.9233	0.9233

Table 16: Training Statistics of `xlm-roberta-base_w_context`

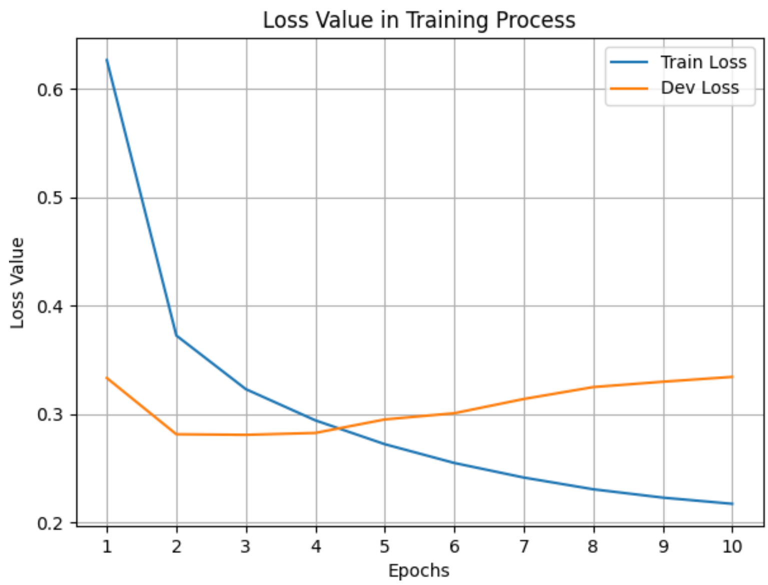

Figure 31: Training Loss of `xlm-roberta-base_w_context`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.620609	0.362943	0.8930	0.8930	0.8930	0.8930
2	0.372899	0.322077	0.9118	0.9118	0.9118	0.9118
3	0.316445	0.327105	0.9148	0.9148	0.9148	0.9148
4	0.282124	0.339592	0.9152	0.9152	0.9152	0.9152
5	0.256676	0.379336	0.9164	0.9164	0.9164	0.9164
6	0.237119	0.389280	0.9168	0.9168	0.9168	0.9168
7	0.221237	0.412443	0.9177	0.9177	0.9177	0.9177
8	0.209032	0.440281	0.9178	0.9178	0.9178	0.9178
9	0.200236	0.461594	0.9172	0.9172	0.9172	0.9172
10	0.193460	0.473301	0.9178	0.9178	0.9178	0.9178

Table 17: Training Statistics of `xlm-roberta-base_o_context`

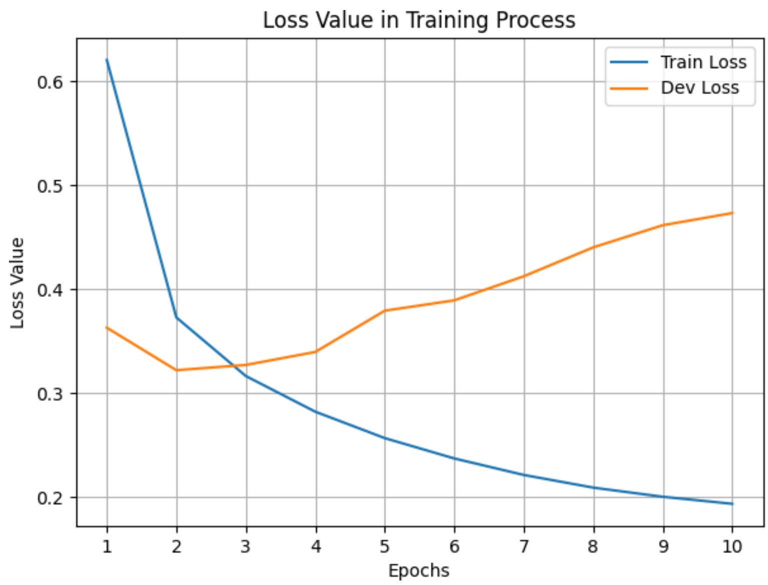

Figure 32: Training Loss of `xlm-roberta-base_o_context`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.244420	0.329268	0.9098	0.9098	0.9098	0.9098
2	0.255996	0.341756	0.9120	0.9120	0.9120	0.9120
3	0.243646	0.343102	0.9120	0.9120	0.9120	0.9120
4	0.233841	0.355693	0.9123	0.9123	0.9123	0.9123
5	0.224557	0.380317	0.9108	0.9108	0.9108	0.9108
6	0.217741	0.377650	0.9118	0.9118	0.9118	0.9118
7	0.210668	0.390299	0.9118	0.9118	0.9118	0.9118
8	0.205027	0.387390	0.9115	0.9115	0.9115	0.9115
9	0.200549	0.396471	0.9113	0.9113	0.9113	0.9113
10	0.196795	0.404745	0.9111	0.9111	0.9111	0.9111

Table 18: Training Statistics of `xlm-roberta-base_w_context_2_DE_sents`

Figure 33: Training Loss of `xlm-roberta-base_w_context_2_DE_sents`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.218628	0.322156	0.9174	0.9174	0.9174	0.9174
2	0.227570	0.338373	0.9204	0.9204	0.9204	0.9204
3	0.217067	0.357133	0.9195	0.9195	0.9195	0.9195
4	0.206302	0.358275	0.9200	0.9200	0.9200	0.9200
5	0.197445	0.373015	0.9207	0.9207	0.9207	0.9207
6	0.189724	0.384272	0.9192	0.9192	0.9192	0.9192
7	0.183825	0.395361	0.9207	0.9207	0.9207	0.9207
8	0.178586	0.399560	0.9199	0.9199	0.9199	0.9199
9	0.173985	0.402610	0.9201	0.9201	0.9201	0.9201
10	0.170116	0.408200	0.9202	0.9202	0.9202	0.9202

Table 19: Training Statistics of `xlm-roberta-base_w_context_2_EN-US_sents`

Figure 34: Training Loss of `xlm-roberta-base_w_context_2_EN-US_sents`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.209790	0.302707	0.9245	0.9245	0.9245	0.9245
2	0.218609	0.310268	0.9288	0.9288	0.9288	0.9288
3	0.207455	0.326701	0.9276	0.9276	0.9276	0.9276
4	0.196017	0.330815	0.9275	0.9275	0.9275	0.9275
5	0.189339	0.343319	0.9301	0.9301	0.9301	0.9301
6	0.181274	0.360828	0.9289	0.9289	0.9289	0.9289
7	0.175659	0.372735	0.9281	0.9281	0.9281	0.9281
8	0.169603	0.379076	0.9287	0.9287	0.9287	0.9287
9	0.165134	0.391918	0.9290	0.9290	0.9290	0.9290
10	0.163075	0.395354	0.9288	0.9288	0.9288	0.9288

Table 20: Training Statistics of `xlm-roberta-base_w_context_2_FR_sents`

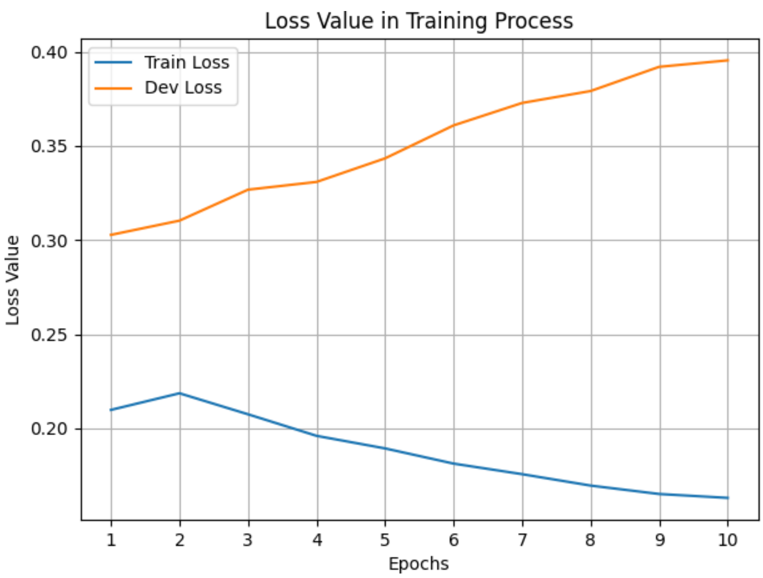

Figure 35: Training Loss of `xlm-roberta-base_w_context_2_FR_sents`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.210239	0.312508	0.9245	0.9245	0.9245	0.9245
2	0.217143	0.312460	0.9267	0.9267	0.9267	0.9267
3	0.205199	0.332618	0.9275	0.9275	0.9275	0.9275
4	0.194661	0.346195	0.9291	0.9291	0.9291	0.9291
5	0.186977	0.361952	0.9287	0.9287	0.9287	0.9287
6	0.179208	0.366749	0.9281	0.9281	0.9281	0.9281
7	0.172972	0.373735	0.9282	0.9282	0.9282	0.9282
8	0.167097	0.378889	0.9288	0.9288	0.9288	0.9288
9	0.163712	0.386165	0.9285	0.9285	0.9285	0.9285
10	0.159562	0.389100	0.9286	0.9286	0.9286	0.9286

Table 21: Training Statistics of `xlm-roberta-base_w_context_2_IT_sents`

Figure 36: Training Loss of `xlm-roberta-base_w_context_2_IT_sents`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.731294	0.322967	0.8981	0.8981	0.8981	0.8981
2	0.394313	0.271130	0.9155	0.9155	0.9155	0.9155
3	0.346800	0.260077	0.9193	0.9193	0.9193	0.9193
4	0.321175	0.257347	0.9211	0.9211	0.9211	0.9211
5	0.303478	0.257359	0.9219	0.9219	0.9219	0.9219
6	0.290428	0.260759	0.9227	0.9227	0.9227	0.9227
7	0.279794	0.263402	0.9235	0.9235	0.9235	0.9235
8	0.272484	0.261250	0.9239	0.9239	0.9239	0.9239
9	0.266799	0.265588	0.9239	0.9239	0.9239	0.9239
10	0.263390	0.267697	0.9241	0.9241	0.9241	0.9241

Table 22: Training Statistics of `xlm-roberta-base_o_context_job`

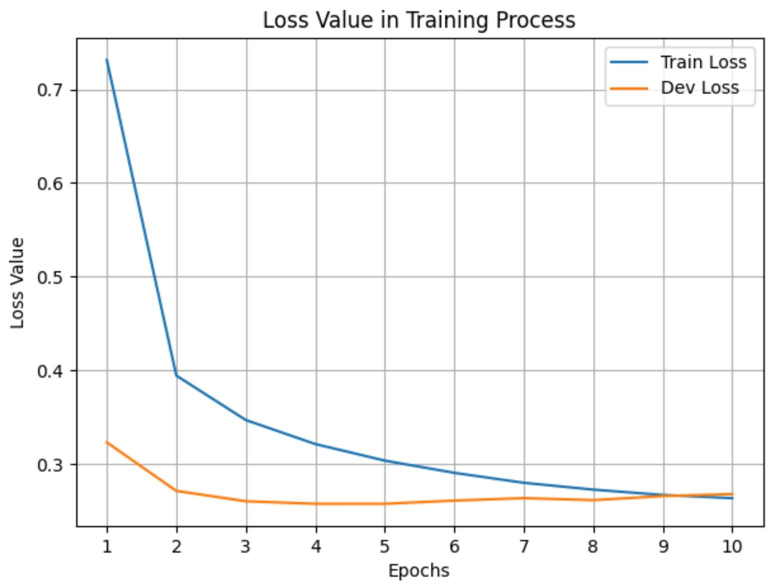

Figure 37: Training Loss of `xlm-roberta-base_o_context_job`

EPOCH	TRAIN_ LOSS	DEV_ LOSS	DEV_ PRECISION	DEV_ RECALL	DEV_ F1	DEV_ ACCURACY
1	0.737120	0.308531	0.9019	0.9019	0.9019	0.9019
2	0.389067	0.267079	0.9161	0.9161	0.9161	0.9161
3	0.346632	0.256546	0.9197	0.9197	0.9197	0.9197
4	0.322796	0.256541	0.9217	0.9217	0.9217	0.9217
5	0.306024	0.252527	0.9233	0.9233	0.9233	0.9233
6	0.293550	0.251217	0.9242	0.9242	0.9242	0.9242
7	0.284225	0.253111	0.9246	0.9246	0.9246	0.9246
8	0.276510	0.257150	0.9250	0.9250	0.9250	0.9250
9	0.271723	0.257902	0.9251	0.9251	0.9251	0.9251
10	0.269010	0.259727	0.9251	0.9251	0.9251	0.9251

Table 23: Training Statistics of `xlm-roberta-base-job`

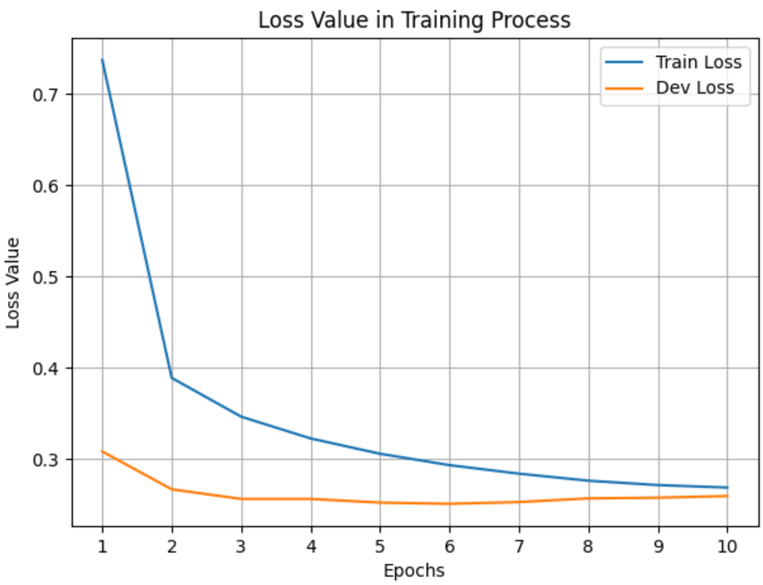

Figure 38: Training Loss of `xlm-roberta-base-job`

C Experiment Results of Trained Models

This appendix contains the evaluation results and confusion matrices of trained sequence labeling models.

C.1 `bert-base-multilingual-cased`

```
Test set: silver
Language: EN-US
- F-score (micro) 0.9218
- F-score (macro) 0.8936
- Accuracy 0.9218

By class:
              precision    recall  f1-score   support

          60     0.9234    0.9252    0.9243     44085
          30     0.9458    0.9267    0.9362     33472
          10     0.9044    0.9201    0.9122     23052
          70     0.9338    0.9395    0.9366     17932
          80     0.8911    0.9028    0.8969     12569
          50     0.8682    0.8347    0.8511      2589
          40     0.8113    0.8597    0.8348      1005
          20     0.8821    0.8325    0.8566       764

    accuracy                         0.9218    135468
   macro avg     0.8950    0.8926    0.8936    135468
weighted avg     0.9220    0.9218    0.9218    135468

Test set: silver
```

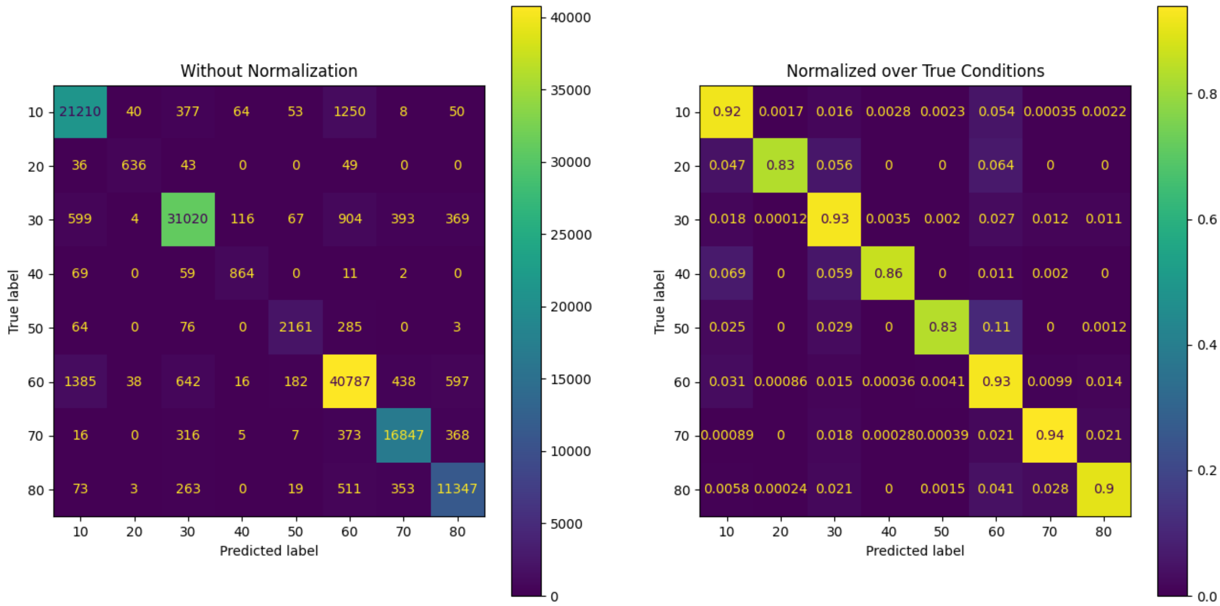

Figure 39: Confusion Matrix of EN-US_silver of `bert-base-multilingual-cased`

```
Language: FR
- F-score (micro) 0.9244
- F-score (macro) 0.8825
- Accuracy 0.9244

By class:
              precision    recall  f1-score   support

          60     0.9272    0.9297    0.9285     51052
          30     0.9499    0.9288    0.9392     34599
          10     0.9090    0.9261    0.9175     25856
          70     0.9403    0.9441    0.9422     20579
          80     0.8866    0.9004    0.8934     14964
          50     0.8683    0.8558    0.8620      3051
          40     0.8032    0.7668    0.7846      1102
          20     0.8389    0.7516    0.7929       797

    accuracy                         0.9244    152000
   macro avg     0.8904    0.8754    0.8825    152000
weighted avg     0.9245    0.9244    0.9244    152000
```

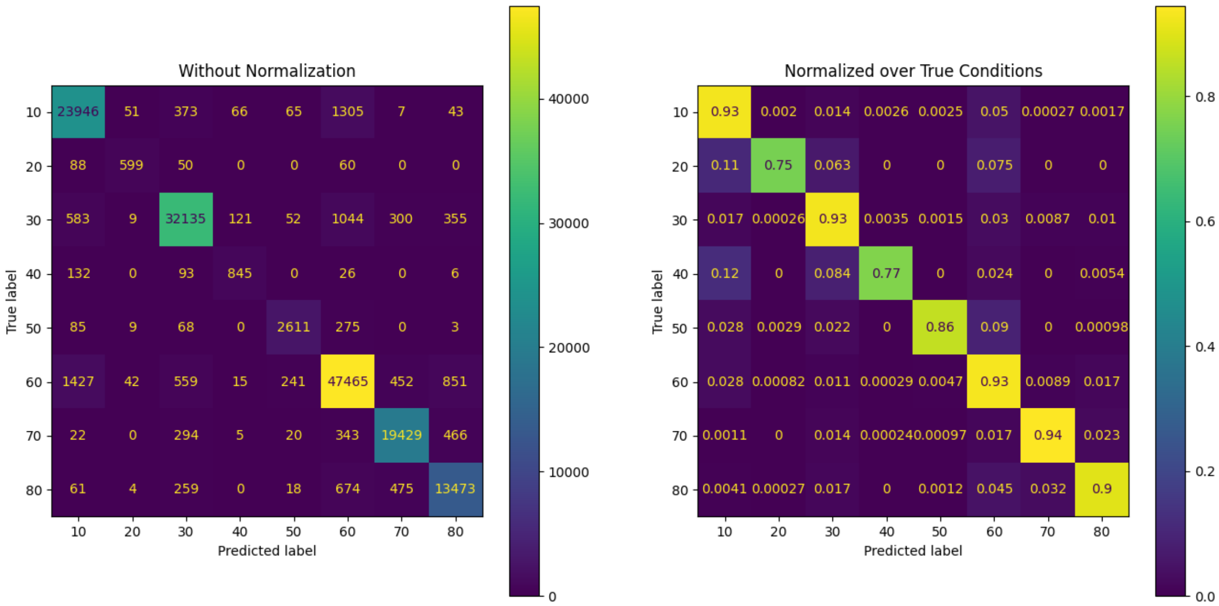

Figure 40: Confusion Matrix of FR_silver of `bert-base-multilingual-cased`

```
Test set: silver
Language: IT
- F-score (micro) 0.9214
- F-score (macro) 0.8835
- Accuracy 0.9214

By class:
```

	precision	recall	f1-score	support
60	0.9225	0.9249	0.9237	45968
30	0.9503	0.9321	0.9411	33113
10	0.8978	0.9189	0.9082	24143
70	0.9407	0.9357	0.9382	17926
80	0.8956	0.9029	0.8993	13569
50	0.8477	0.8313	0.8394	2786
40	0.8023	0.8230	0.8126	1085
20	0.8339	0.7789	0.8055	683
accuracy			0.9214	139273
macro avg	0.8863	0.8810	0.8835	139273
weighted avg	0.9217	0.9214	0.9215	139273

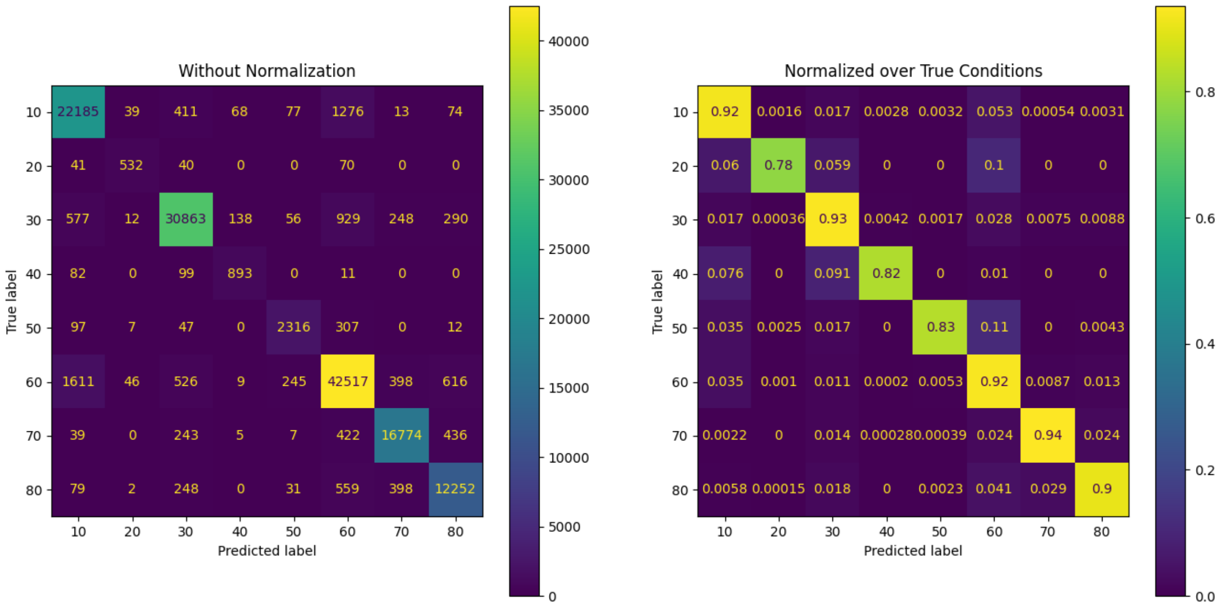

Figure 41: Confusion Matrix of IT_silver of `bert-base-multilingual-cased`

```
Test set: gold
Language: EN-US
- F-score (micro) 0.942
- F-score (macro) 0.8204
- Accuracy 0.942

By class:
              precision    recall  f1-score   support

          60     0.9440    0.9800    0.9617      3597
          10     0.9582    0.8971    0.9266      1506
          30     0.9718    0.8478    0.9056       854
          70     0.9366    0.9718    0.9539       745
          80     0.8925    0.9213    0.9067       559
          50     0.8333    1.0000    0.9091        70
          40     0.0000    0.0000    0.0000         4
          20     1.0000    1.0000    1.0000         7

    accuracy                         0.9420      7342
   macro avg     0.8171    0.8272    0.8204      7342
weighted avg     0.9440    0.9420    0.9420      7342
```

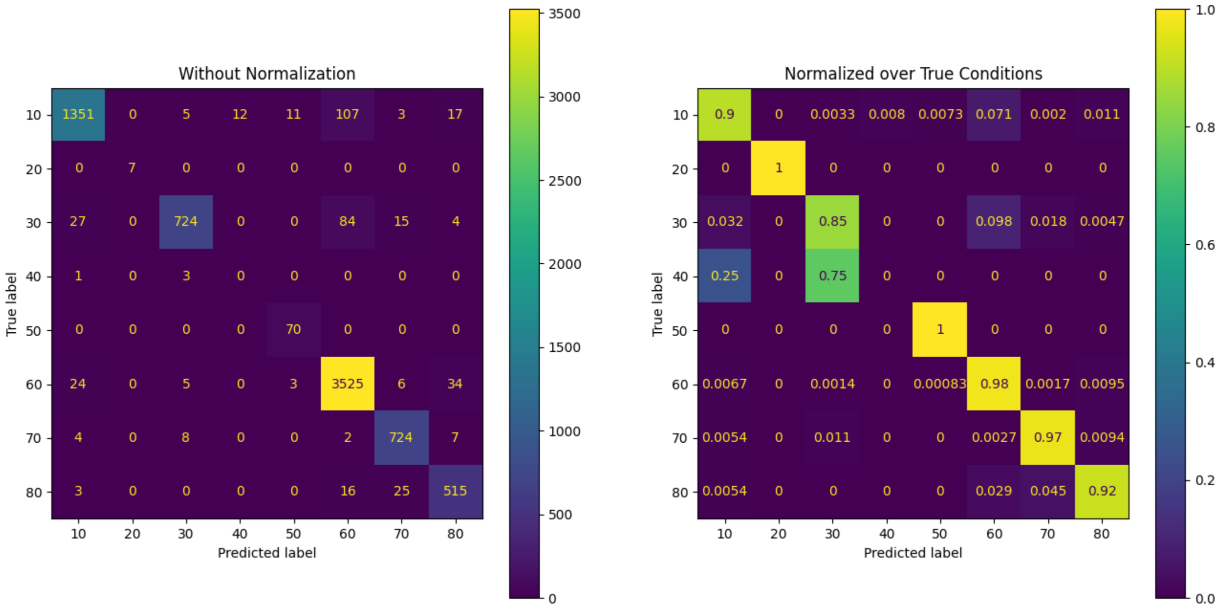

Figure 42: Confusion Matrix of EN-US_gold of `bert-base-multilingual-cased`

```
Test set: gold
Language: FR
- F-score (micro) 0.9517
- F-score (macro) 0.9067
- Accuracy 0.9517

By class:
              precision    recall  f1-score   support

          60     0.9455    0.9656    0.9555      2499
          30     0.9811    0.9419    0.9611      1653
          70     0.9877    0.9223    0.9539      1223
          10     0.9125    0.9858    0.9478      1058
          80     0.9406    0.9582    0.9493       958
          50     0.8212    1.0000    0.9018       124
          40     1.0000    0.5172    0.6818        58
          20     1.0000    0.8222    0.9024        45

    accuracy                         0.9517      7618
   macro avg     0.9486    0.8892    0.9067      7618
weighted avg     0.9535    0.9517    0.9513      7618
```

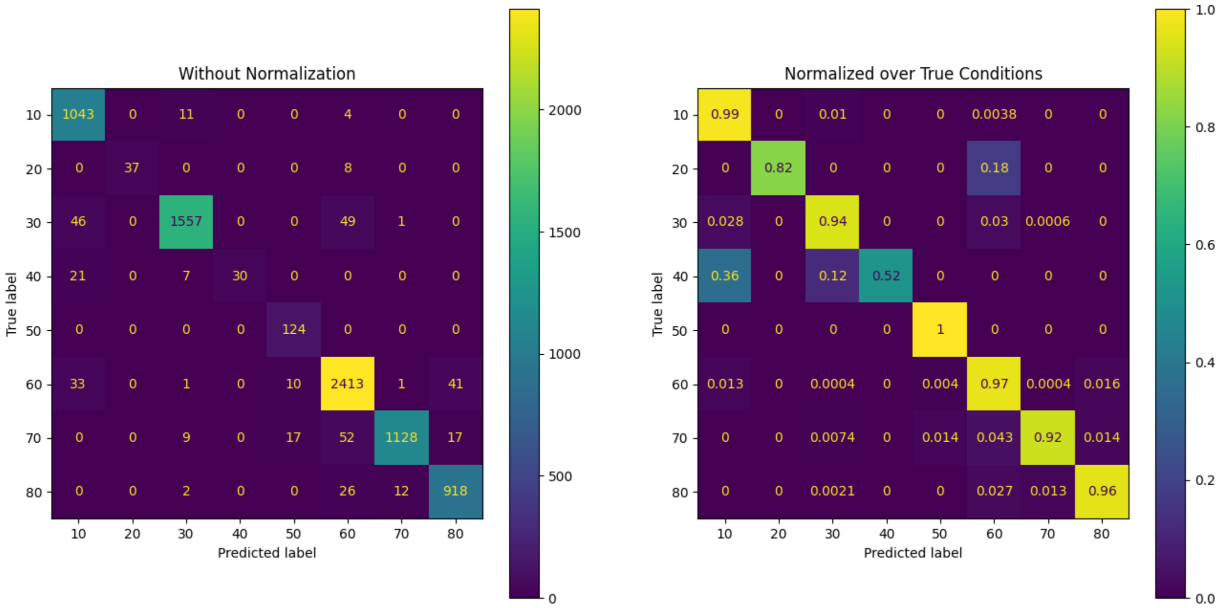

Figure 43: Confusion Matrix of FR_gold of `bert-base-multilingual-cased`

```
Test set: gold
Language: IT
- F-score (micro) 0.9311
- F-score (macro) 0.9135
- Accuracy 0.9311

By class:
              precision    recall  f1-score   support

          30     0.9515    0.9161    0.9334      2442
          60     0.9298    0.9312    0.9305      2077
          10     0.9596    0.9582    0.9589      1461
          70     0.9849    0.9413    0.9626       903
          80     0.8579    0.9538    0.9033       715
          50     0.7378    0.8400    0.7856       325
          40     0.7600    1.0000    0.8636        19
          20     1.0000    0.9412    0.9697        17

    accuracy                         0.9311      7959
   macro avg     0.8977    0.9352    0.9135      7959
weighted avg     0.9336    0.9311    0.9318      7959
```

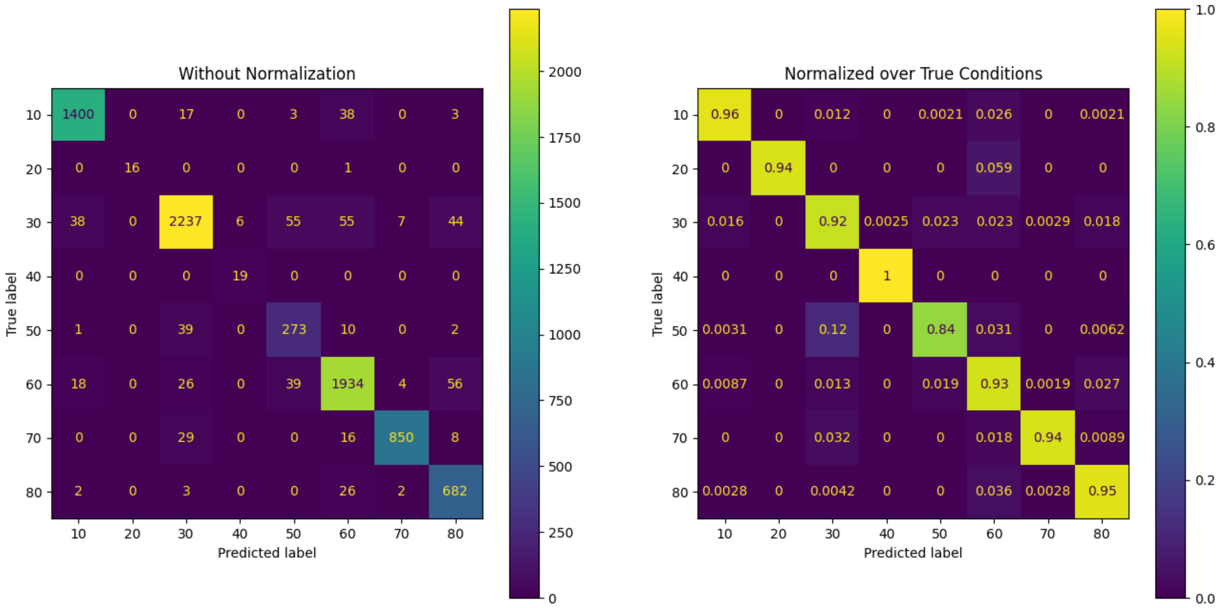

Figure 44: Confusion Matrix of IT_gold of `bert-base-multilingual-cased`

```
Test set: gold
Language: DE
- F-score (micro) 0.9132
- F-score (macro) 0.8706
- Accuracy 0.9132

By class:
```

	precision	recall	f1-score	support
60	0.9193	0.9186	0.9189	38275
30	0.9421	0.9169	0.9294	30976
10	0.9018	0.9284	0.9149	20636
70	0.9157	0.9298	0.9227	15307
80	0.8607	0.8672	0.8640	11003
50	0.8274	0.8299	0.8287	1964
40	0.7949	0.7716	0.7831	924
20	0.8375	0.7714	0.8031	608
accuracy			0.9132	119693
macro avg	0.8749	0.8667	0.8706	119693
weighted avg	0.9135	0.9132	0.9132	119693

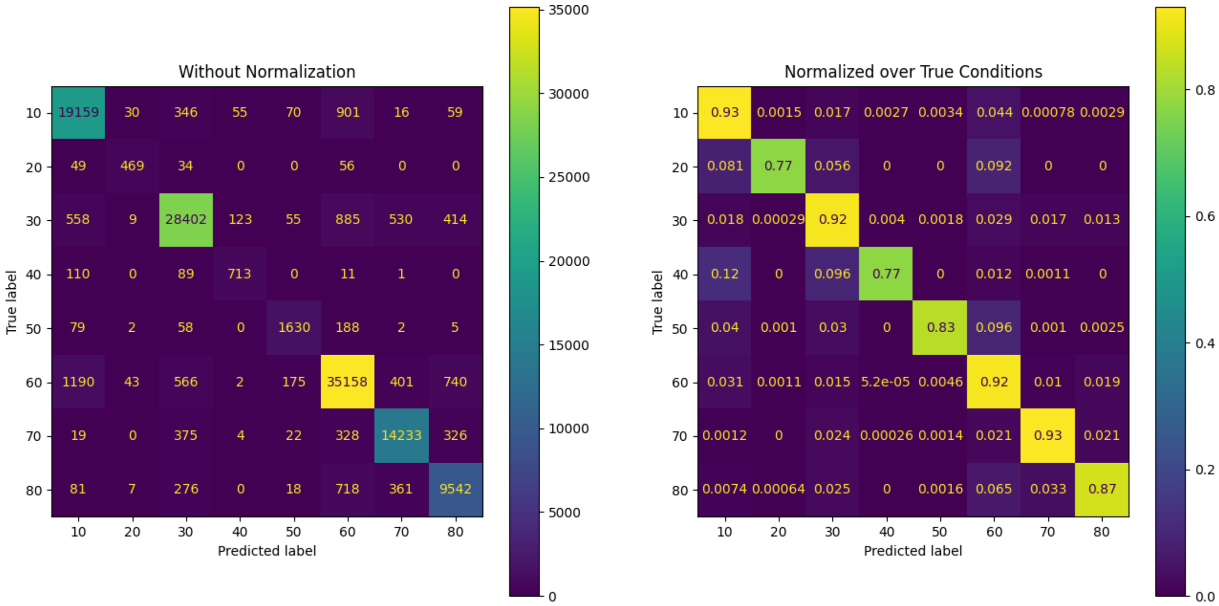

Figure 45: Confusion Matrix of DE_gold of `bert-base-multilingual-cased`

C.2 `bert-base-multilingual-cased_w_context`

```
Test set: silver
Language: EN-US
- F-score (micro) 0.9176
- F-score (macro) 0.8761
- Accuracy 0.9176

By class:
```

	precision	recall	f1-score	support
60	0.9239	0.9226	0.9232	44085
30	0.9448	0.9217	0.9331	33472
10	0.8985	0.9153	0.9068	23052
70	0.9289	0.9377	0.9333	17932
80	0.8739	0.9025	0.8880	12569
50	0.8523	0.8405	0.8464	2589
40	0.7700	0.7662	0.7681	1005
20	0.8672	0.7605	0.8103	764
accuracy			0.9176	135468

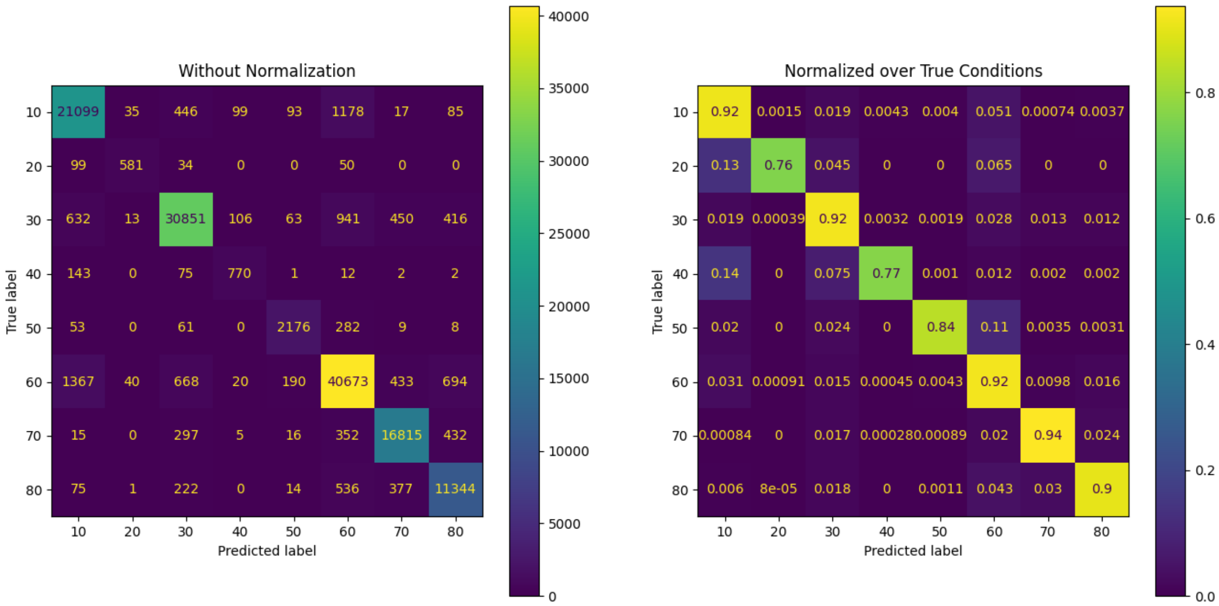

Figure 46: Confusion Matrix of EN-US_silver of `bert-base-multilingual-cased_w_context`

```
   macro avg      0.8824    0.8709    0.8761    135468
weighted avg      0.9179    0.9176    0.9177    135468


Test set: silver
Language: FR
- F-score (micro) 0.9243
- F-score (macro) 0.8762
- Accuracy 0.9243

By class:
              precision    recall  f1-score   support

          60     0.9316    0.9276    0.9296     51052
          30     0.9476    0.9307    0.9391     34599
          10     0.9019    0.9204    0.9110     25856
          70     0.9383    0.9481    0.9431     20579
          80     0.8991    0.9061    0.9026     14964
          50     0.8519    0.8787    0.8651      3051
          40     0.7352    0.7305    0.7328      1102
          20     0.8499    0.7315    0.7862       797
```

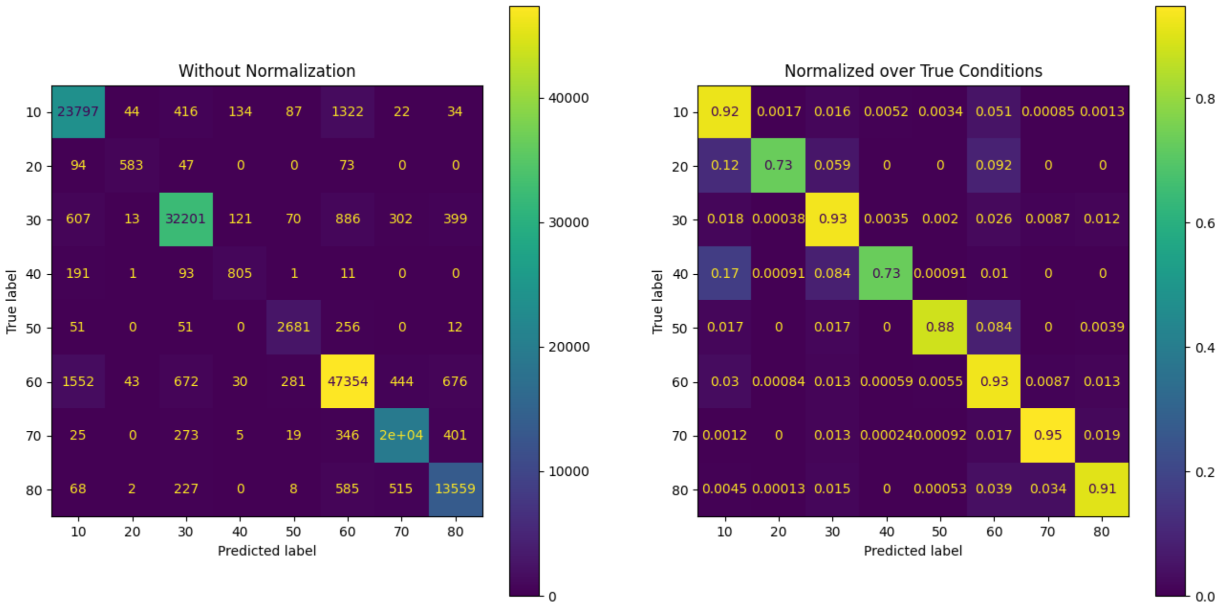

Figure 47: Confusion Matrix of FR_silver of `bert-base-multilingual-cased_w_context`

accuracy			0.9243	152000
macro avg	0.8819	0.8717	0.8762	152000
weighted avg	0.9244	0.9243	0.9243	152000

```
Test set: silver
Language: IT
- F-score (micro) 0.9218
- F-score (macro) 0.8727
- Accuracy 0.9218

By class:
```

	precision	recall	f1-score	support
60	0.9283	0.9223	0.9253	45968
30	0.9508	0.9366	0.9437	33113
10	0.8936	0.9202	0.9067	24143
70	0.9380	0.9382	0.9381	17926
80	0.8960	0.9049	0.9005	13569
50	0.8284	0.8453	0.8367	2786
40	0.7752	0.7502	0.7625	1085

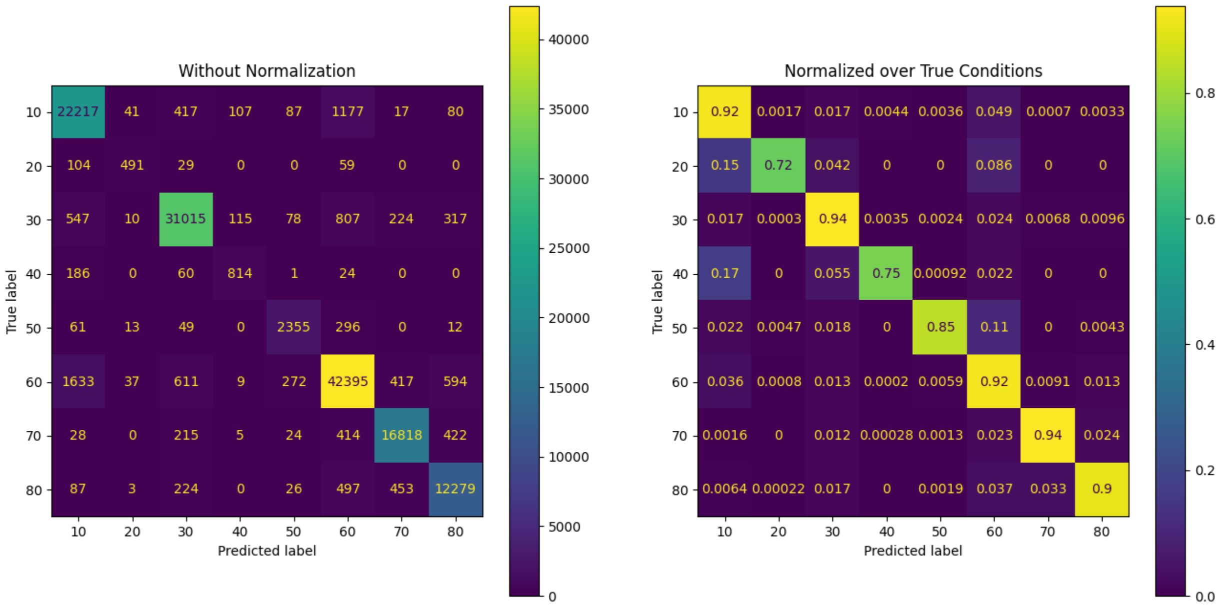

Figure 48: Confusion Matrix of IT_silver of `bert-base-multilingual-cased_w_context`

20	0.8252	0.7189	0.7684	683
accuracy			0.9218	139273
macro avg	0.8794	0.8671	0.8727	139273
weighted avg	0.9220	0.9218	0.9219	139273

Test set: gold
Language: EN-US
- F-score (micro) 0.8909
- F-score (macro) 0.7812
- Accuracy 0.8909

By class:

	precision	recall	f1-score	support
60	0.9403	0.9675	0.9537	3597
10	0.9337	0.7198	0.8129	1506
30	0.9404	0.8314	0.8825	854
70	0.9311	0.9248	0.9279	745
80	0.8776	0.9106	0.8938	559
40	0.0032	0.2500	0.0064	4

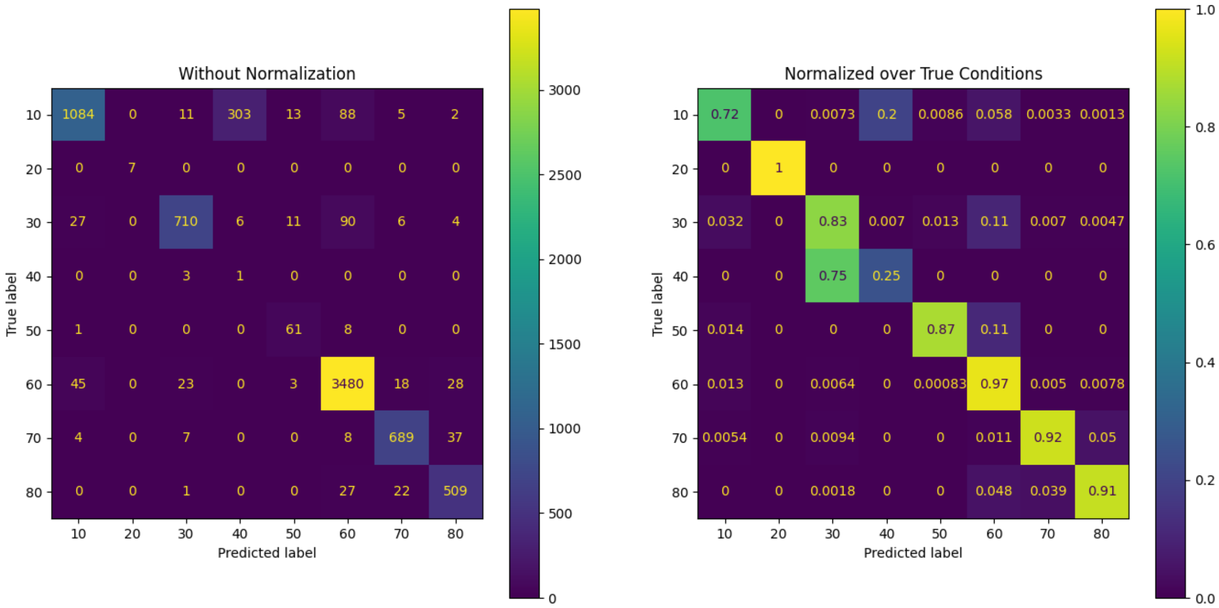

Figure 49: Confusion Matrix of EN-US_gold of `bert-base-multilingual-cased_w_context`

```
          50       0.6932    0.8714    0.7722        70
          20       1.0000    1.0000    1.0000         7

    accuracy                           0.8909      7342
   macro avg       0.7899    0.8094    0.7812      7342
weighted avg       0.9304    0.8909    0.9072      7342
```

```
Test set: gold
Language: FR
- F-score (micro) 0.922
- F-score (macro) 0.8784
- Accuracy 0.922

By class:
              precision    recall  f1-score   support

          60     0.9209    0.9360    0.9284      2499
          30     0.9444    0.9250    0.9346      1653
          70     0.9730    0.9419    0.9572      1223
          10     0.8996    0.8894    0.8945      1058
          80     0.8895    0.9071    0.8982       958
```

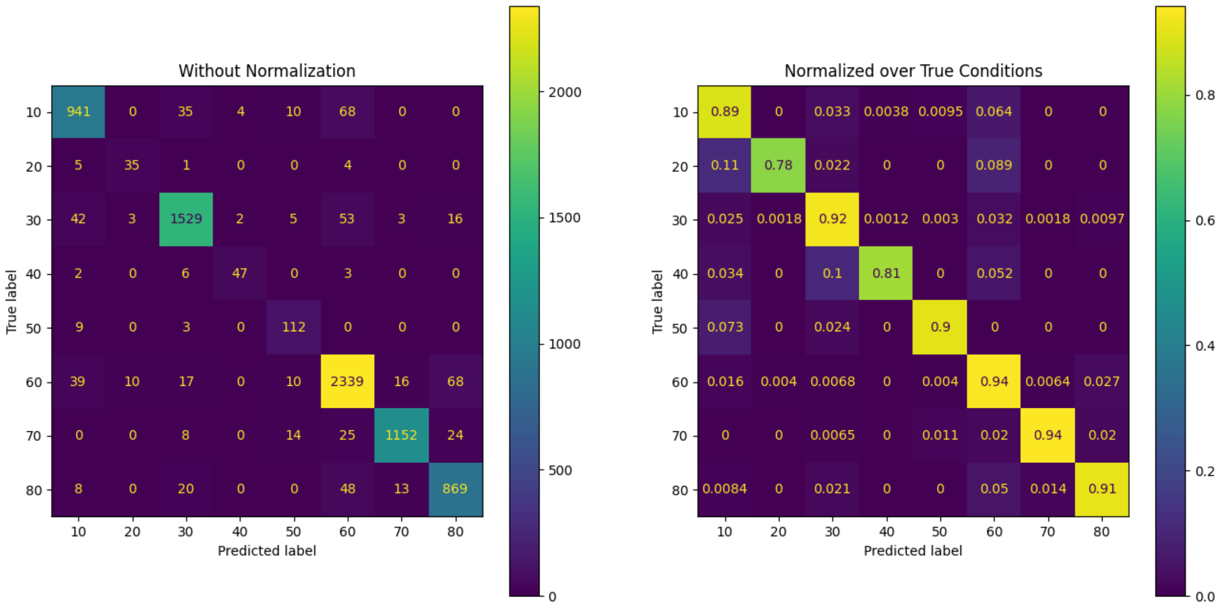

Figure 50: Confusion Matrix of FR_gold of `bert-base-multilingual-cased_w_context`

50	0.7417	0.9032	0.8145	124
40	0.8868	0.8103	0.8468	58
20	0.7292	0.7778	0.7527	45
accuracy			0.9220	7618
macro avg	0.8731	0.8863	0.8784	7618
weighted avg	0.9231	0.9220	0.9223	7618

```
Test set: gold
Language: IT
- F-score (micro) 0.9152
- F-score (macro) 0.8748
- Accuracy 0.9152
```

By class:

	precision	recall	f1-score	support
30	0.9465	0.9128	0.9293	2442
60	0.9128	0.9023	0.9075	2077
10	0.9486	0.9343	0.9414	1461
70	0.9622	0.9291	0.9454	903

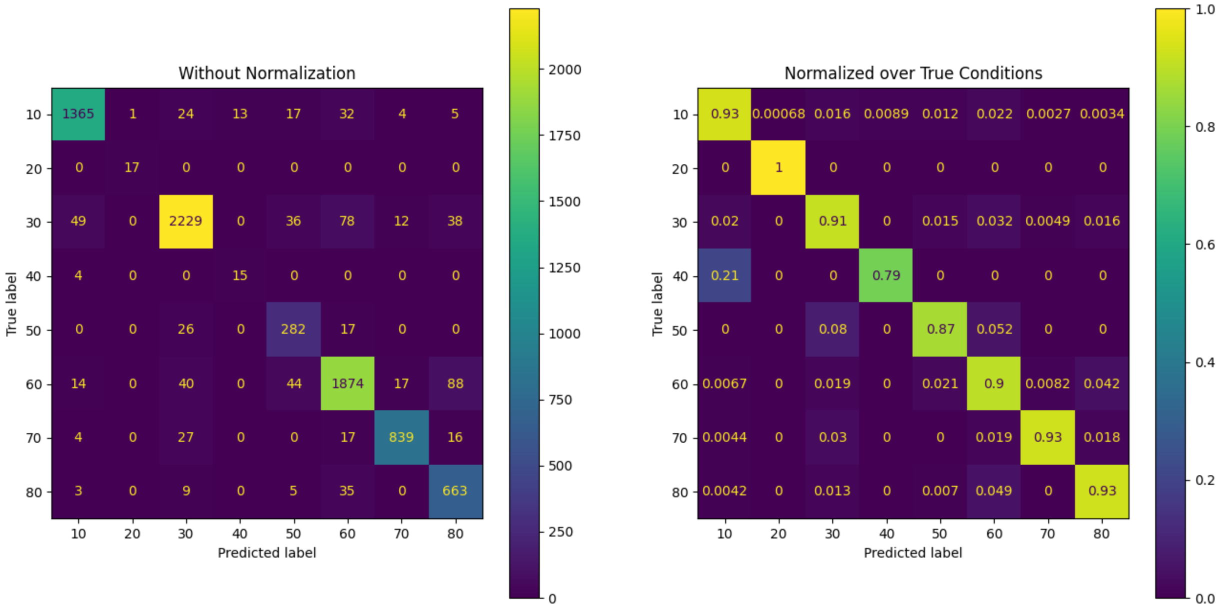

Figure 51: Confusion Matrix of IT_gold of `bert-base-multilingual-cased_w_context`

80	0.8185	0.9273	0.8695	715
50	0.7344	0.8677	0.7955	325
40	0.5357	0.7895	0.6383	19
20	0.9444	1.0000	0.9714	17
accuracy			0.9152	7959
macro avg	0.8504	0.9079	0.8748	7959
weighted avg	0.9187	0.9152	0.9162	7959

```
Test set: gold
Language: DE
- F-score (micro) 0.9118
- F-score (macro) 0.8682
- Accuracy 0.9118
```

By class:

	precision	recall	f1-score	support
60	0.9185	0.9165	0.9175	38275
30	0.9387	0.9172	0.9278	30976
10	0.8965	0.9195	0.9079	20636

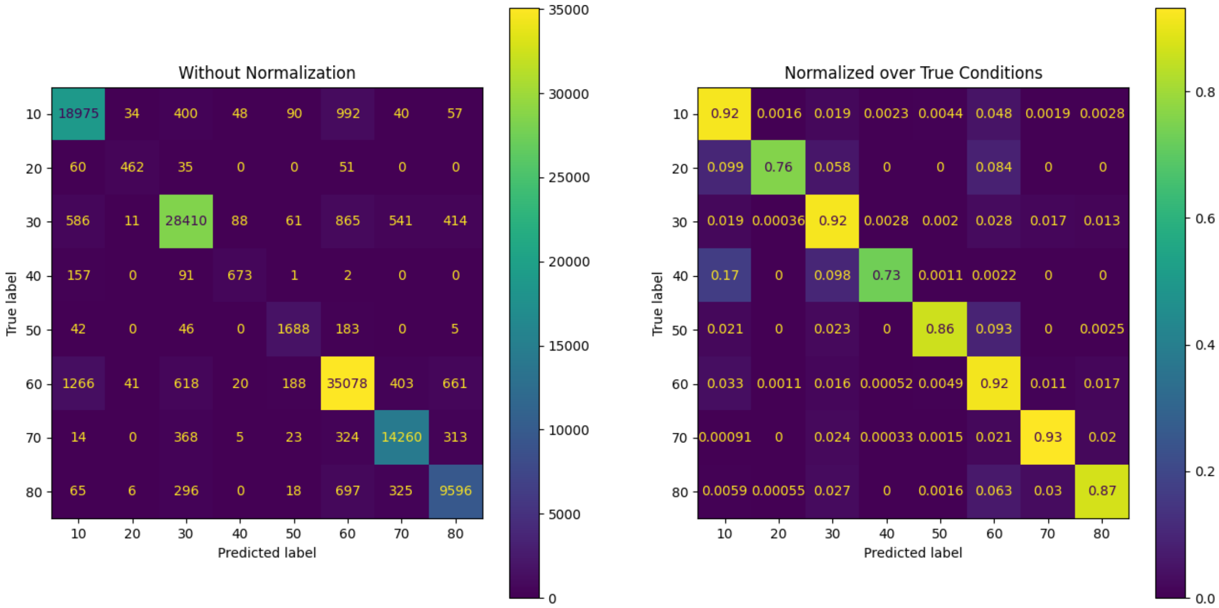

Figure 52: Confusion Matrix of DE_gold of `bert-base-multilingual-cased_w_context`

```
          70      0.9159    0.9316    0.9237     15307
          80      0.8687    0.8721    0.8704     11003
          50      0.8159    0.8595    0.8371      1964
          40      0.8070    0.7284    0.7656       924
          20      0.8339    0.7599    0.7952       608

    accuracy                          0.9118    119693
   macro avg      0.8744    0.8631    0.8682    119693
weighted avg      0.9121    0.9118    0.9119    119693
```

C.3 `bert-base-multilingual-cased_2_DE`

```
Test set: gold
Language: DE
- F-score (micro) 0.9129
- F-score (macro) 0.8697
- Accuracy 0.9129

By class:
```

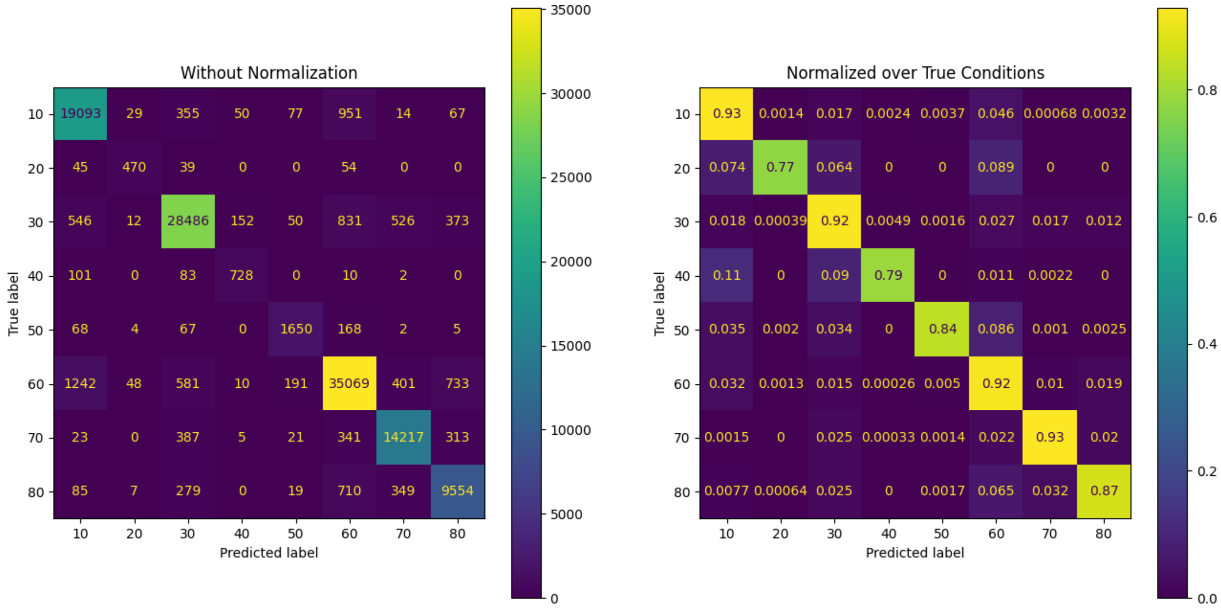

Figure 53: Confusion Matrix of DE_gold of `bert-base-multilingual-cased_2_DE`

	precision	recall	f1-score	support
60	0.9196	0.9162	0.9179	38275
30	0.9408	0.9196	0.9301	30976
10	0.9005	0.9252	0.9127	20636
70	0.9166	0.9288	0.9226	15307
80	0.8650	0.8683	0.8667	11003
50	0.8217	0.8401	0.8308	1964
40	0.7704	0.7879	0.7790	924
20	0.8246	0.7730	0.7980	608
accuracy			0.9129	119693
macro avg	0.8699	0.8699	0.8697	119693
weighted avg	0.9132	0.9129	0.9130	119693

C.4 `bert-base-multilingual-cased_2_EN-US`

```
Test set: silver
Language: EN-US
```

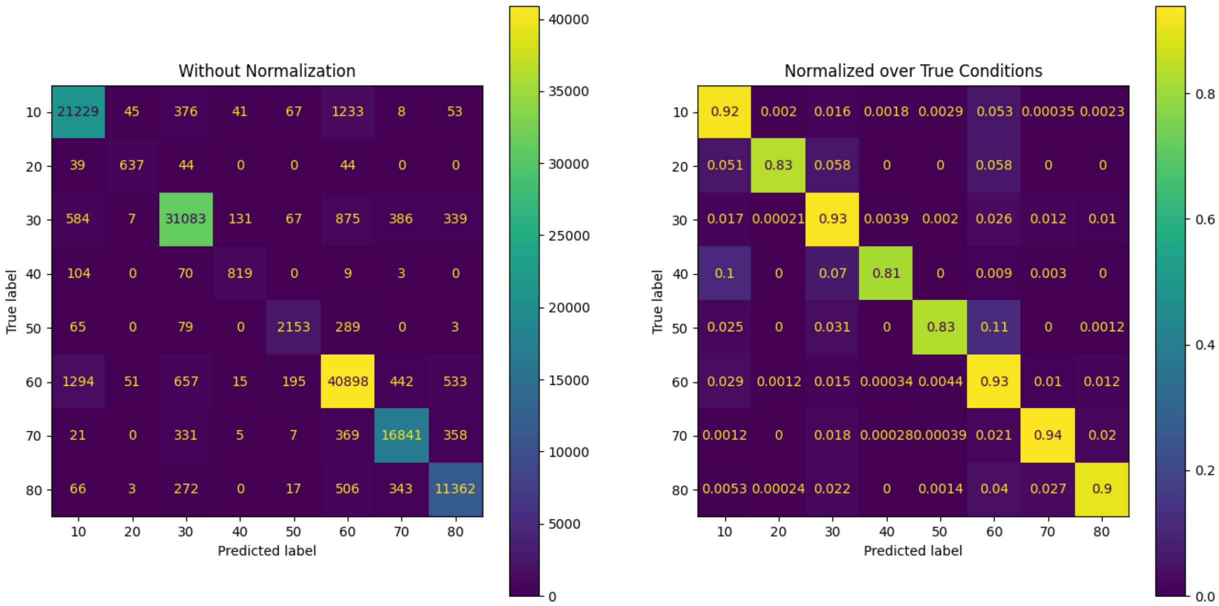

Figure 54: Confusion Matrix of EN-US_silver of `bert-base-multilingual-cased_2_EN-US`

```
- F-score (micro) 0.9229
- F-score (macro) 0.8897
- Accuracy 0.9229

By class:
              precision    recall  f1-score   support

          60     0.9248    0.9277    0.9263     44085
          30     0.9444    0.9286    0.9365     33472
          10     0.9071    0.9209    0.9140     23052
          70     0.9344    0.9392    0.9368     17932
          80     0.8983    0.9040    0.9011     12569
          50     0.8591    0.8316    0.8451      2589
          40     0.8101    0.8149    0.8125      1005
          20     0.8573    0.8338    0.8454       764

    accuracy                         0.9229    135468
   macro avg     0.8920    0.8876    0.8897    135468
weighted avg     0.9230    0.9229    0.9229    135468
```

Test set: gold
Language: EN-US
- F-score (micro) 0.9401
- F-score (macro) 0.8215
- Accuracy 0.9401

By class:

	precision	recall	f1-score	support
60	0.9430	0.9803	0.9613	3597
10	0.9593	0.8911	0.9239	1506
30	0.9510	0.8642	0.9055	854
70	0.9421	0.9611	0.9515	745
80	0.8811	0.9016	0.8912	559
50	0.8961	0.9857	0.9388	70
40	0.0000	0.0000	0.0000	4
20	1.0000	1.0000	1.0000	7
accuracy			0.9401	7342
macro avg	0.8216	0.8230	0.8215	7342
weighted avg	0.9416	0.9401	0.9401	7342

C.5 bert-base-multilingual-cased_2_FR

Test set: silver
Language: FR
- F-score (micro) 0.9255
- F-score (macro) 0.8842
- Accuracy 0.9255

By class:

	precision	recall	f1-score	support
60	0.9289	0.9316	0.9302	51052
30	0.9488	0.9294	0.9390	34599
10	0.9120	0.9247	0.9183	25856
70	0.9390	0.9444	0.9417	20579

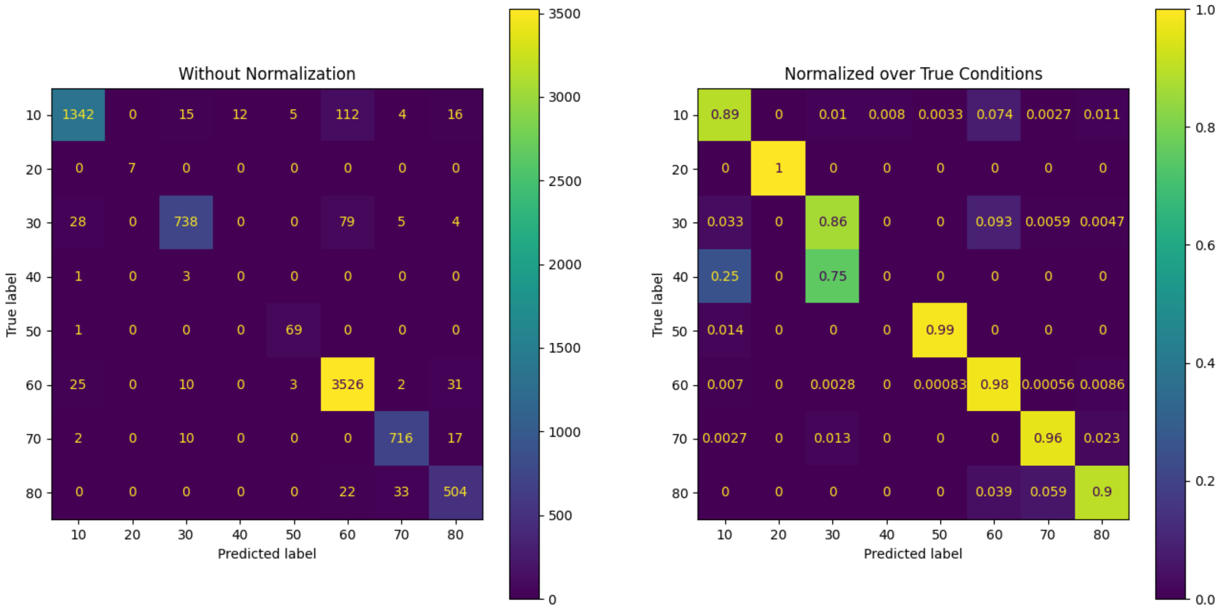

Figure 55: Confusion Matrix of EN-US_gold of `bert-base-multilingual-cased_2_EN-US`

```
           80      0.8908    0.9045    0.8976     14964
           50      0.8658    0.8709    0.8683      3051
           40      0.8067    0.7423    0.7732      1102
           20      0.8506    0.7641    0.8050       797

     accuracy                          0.9255    152000
    macro avg      0.8928    0.8765    0.8842    152000
 weighted avg      0.9256    0.9255    0.9255    152000


Test set: gold
Language: FR
- F-score (micro) 0.95
- F-score (macro) 0.9073
- Accuracy 0.95

By class:
              precision    recall  f1-score   support

           60      0.9478    0.9660    0.9568      2499
           30      0.9712    0.9383    0.9545      1653
           70      0.9827    0.9305    0.9559      1223
```

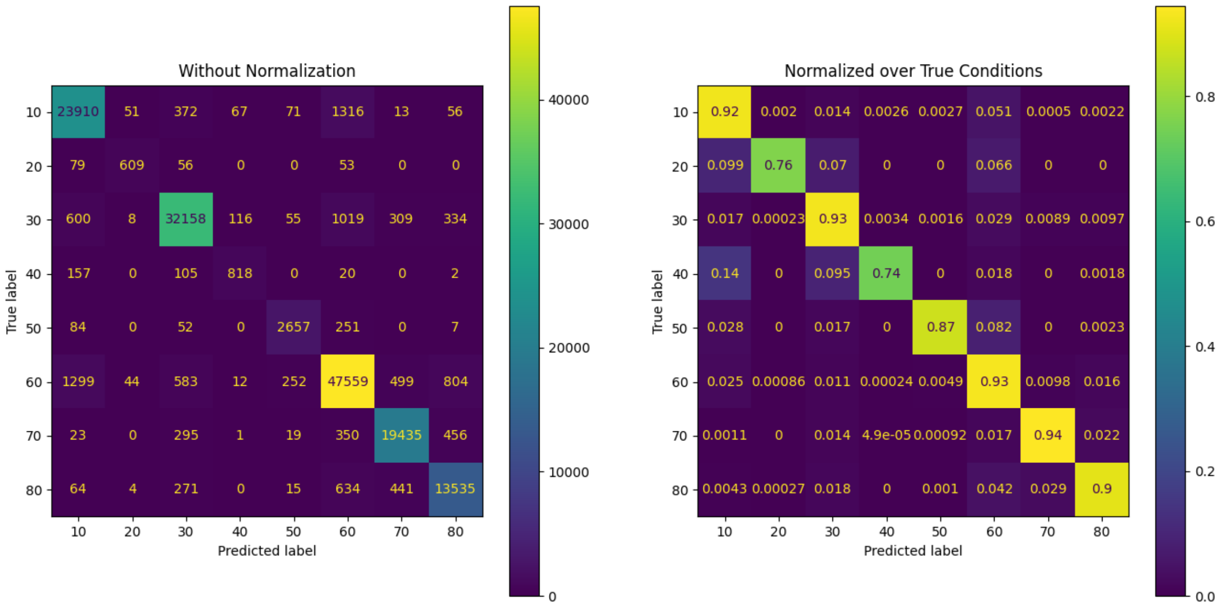

Figure 56: Confusion Matrix of FR_silver of `bert-base-multilingual-cased_2_FR`

10	0.9176	0.9792	0.9474	1058
80	0.9380	0.9468	0.9423	958
50	0.8079	0.9839	0.8873	124
40	1.0000	0.5690	0.7253	58
20	1.0000	0.8000	0.8889	45
accuracy			0.9500	7618
macro avg	0.9457	0.8892	0.9073	7618
weighted avg	0.9515	0.9500	0.9497	7618

C.6 `bert-base-multilingual-cased_2_IT`

```
Test set: silver
Language: IT
- F-score (micro) 0.9235
- F-score (macro) 0.8875
- Accuracy 0.9235

By class:
              precision    recall  f1-score   support
```

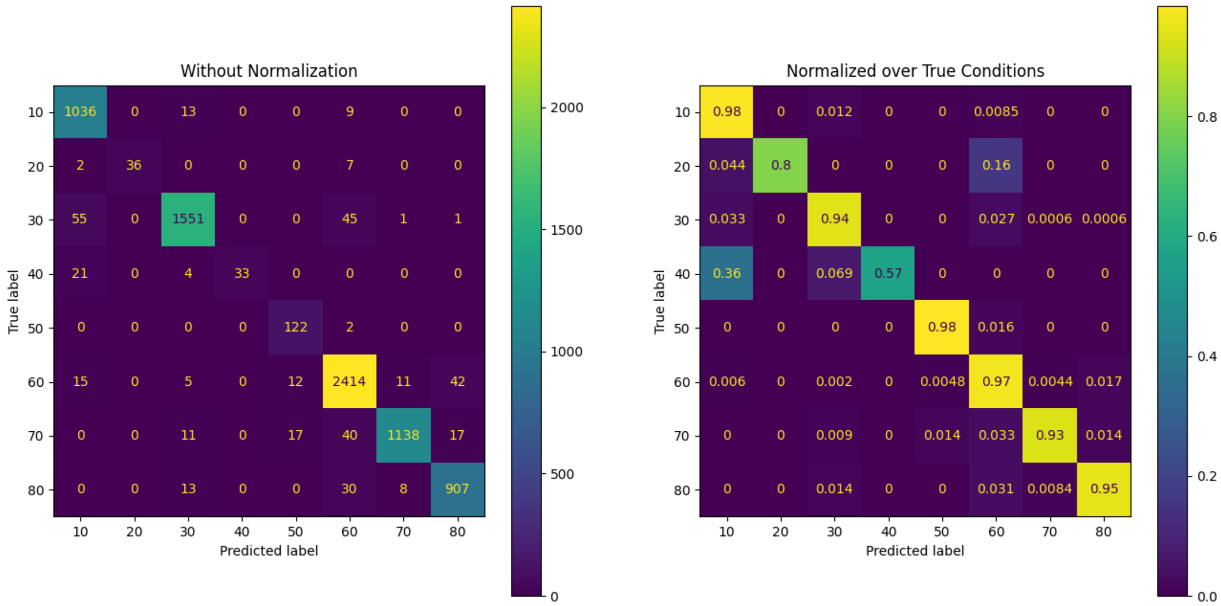

Figure 57: Confusion Matrix of FR_gold of `bert-base-multilingual-cased_2_FR`

60	0.9259	0.9283	0.9271	45968
30	0.9497	0.9324	0.9409	33113
10	0.9037	0.9203	0.9119	24143
70	0.9419	0.9341	0.9380	17926
80	0.8948	0.9112	0.9029	13569
50	0.8301	0.8295	0.8298	2786
40	0.8343	0.8304	0.8323	1085
20	0.8497	0.7862	0.8167	683
accuracy			0.9235	139273
macro avg	0.8913	0.8840	0.8875	139273
weighted avg	0.9237	0.9235	0.9236	139273

```
Test set: gold
Language: IT
- F-score (micro) 0.9257
- F-score (macro) 0.9098
- Accuracy 0.9257

By class:
```

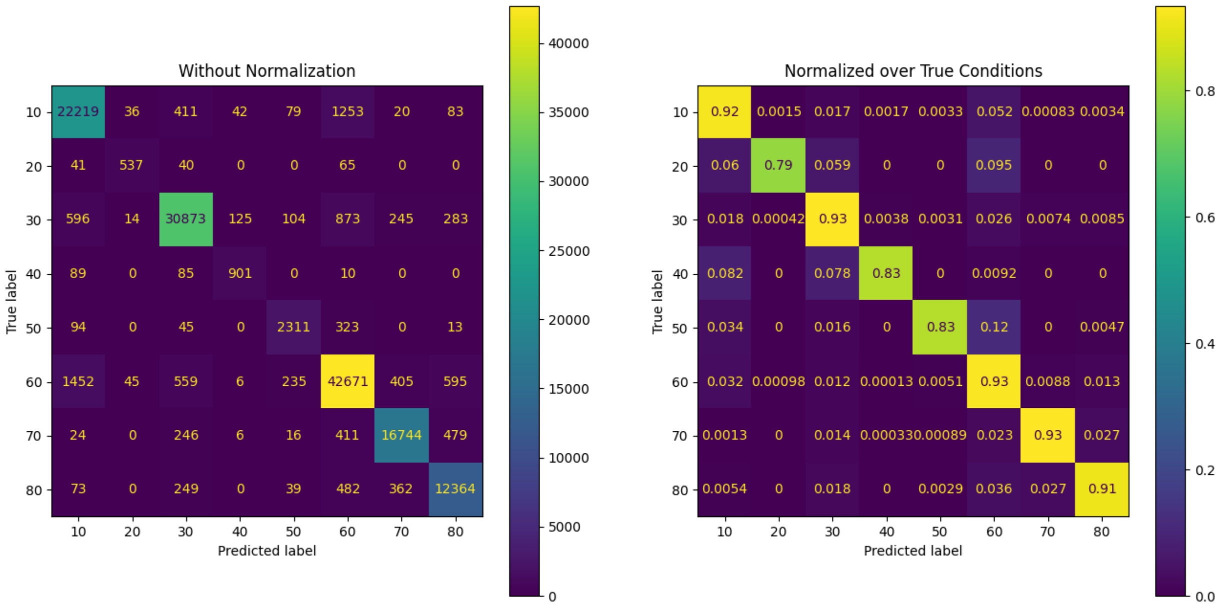

Figure 58: Confusion Matrix of IT_silver of `bert-base-multilingual-cased_2_IT`

	precision	recall	f1-score	support
30	0.9495	0.9087	0.9286	2442
60	0.9167	0.9331	0.9248	2077
10	0.9589	0.9411	0.9499	1461
70	0.9760	0.9457	0.9606	903
80	0.8606	0.9413	0.8991	715
50	0.7287	0.8431	0.7817	325
40	0.7600	1.0000	0.8636	19
20	1.0000	0.9412	0.9697	17
accuracy			0.9257	7959
macro avg	0.8938	0.9318	0.9098	7959
weighted avg	0.9283	0.9257	0.9265	7959

C.7 jobad_bert_finetune_multi

Test set: silver

Language: EN-US

- F-score (micro) 0.9111

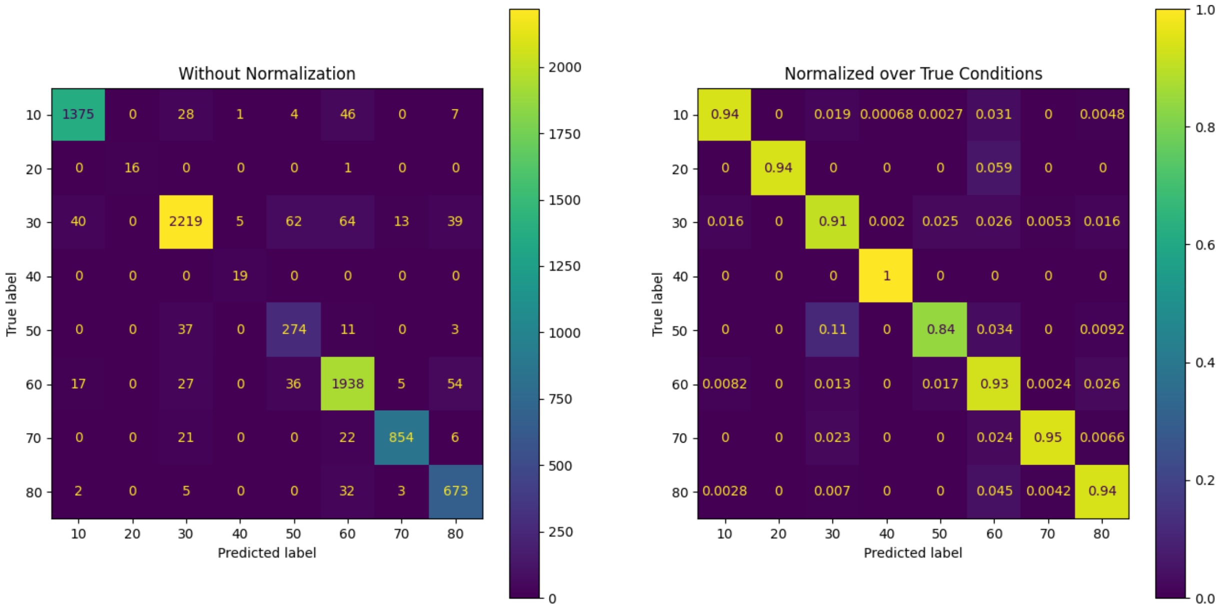

Figure 59: Confusion Matrix of IT_gold of `bert-base-multilingual-cased_2_IT`

```
- F-score (macro) 0.8676
- Accuracy 0.9111

By class:
              precision    recall  f1-score   support

          60     0.9122    0.9211    0.9166     44085
          30     0.9358    0.9224    0.9290     33472
          10     0.8983    0.9120    0.9051     23052
          70     0.9281    0.9252    0.9266     17932
          80     0.8619    0.8749    0.8684     12569
          50     0.8666    0.7756    0.8186      2589
          40     0.8109    0.7423    0.7751      1005
          20     0.8492    0.7592    0.8017       764

    accuracy                         0.9111    135468
   macro avg     0.8829    0.8541    0.8676    135468
weighted avg     0.9111    0.9111    0.9110    135468

Test set: silver
Language: FR
```

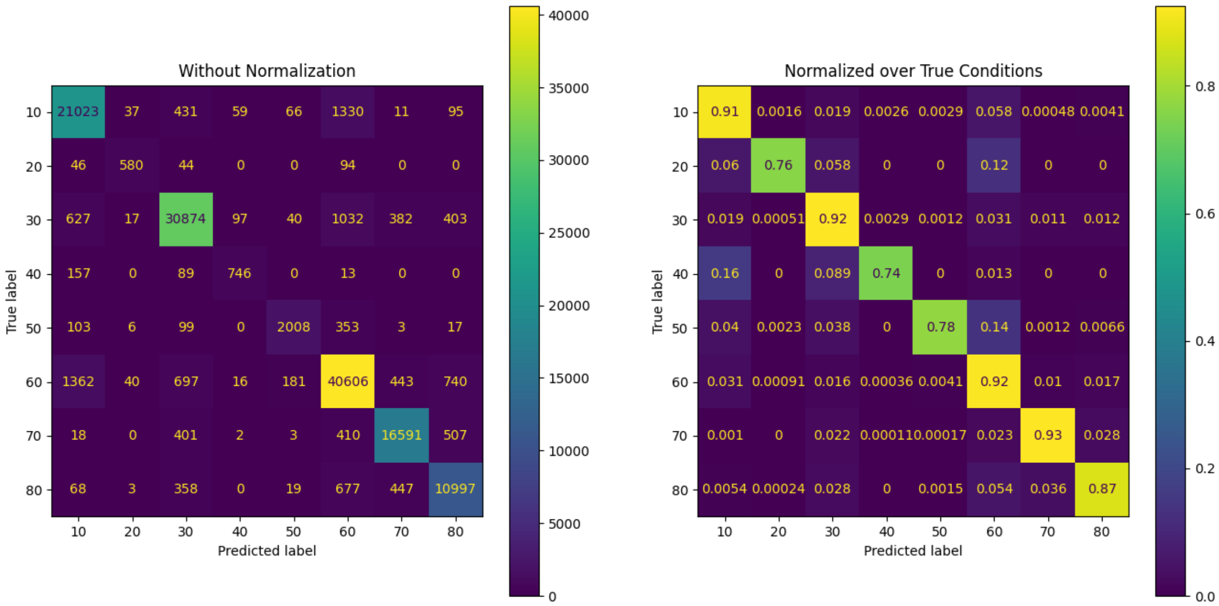

Figure 60: Confusion Matrix of EN-US_silver of `jobad_bert_finetune_multi`

```
- F-score (micro) 0.9054
- F-score (macro) 0.867
- Accuracy 0.9054

By class:
              precision    recall  f1-score   support

          60     0.9065    0.9184    0.9124     51052
          30     0.9354    0.9178    0.9265     34599
          10     0.9054    0.9053    0.9054     25856
          70     0.9117    0.9106    0.9112     20579
          80     0.8483    0.8595    0.8538     14964
          50     0.8343    0.8122    0.8231      3051
          40     0.8292    0.8194    0.8243      1102
          20     0.8310    0.7340    0.7795       797

    accuracy                         0.9054    152000
   macro avg     0.8752    0.8597    0.8670    152000
weighted avg     0.9055    0.9054    0.9054    152000

Test set: silver
```

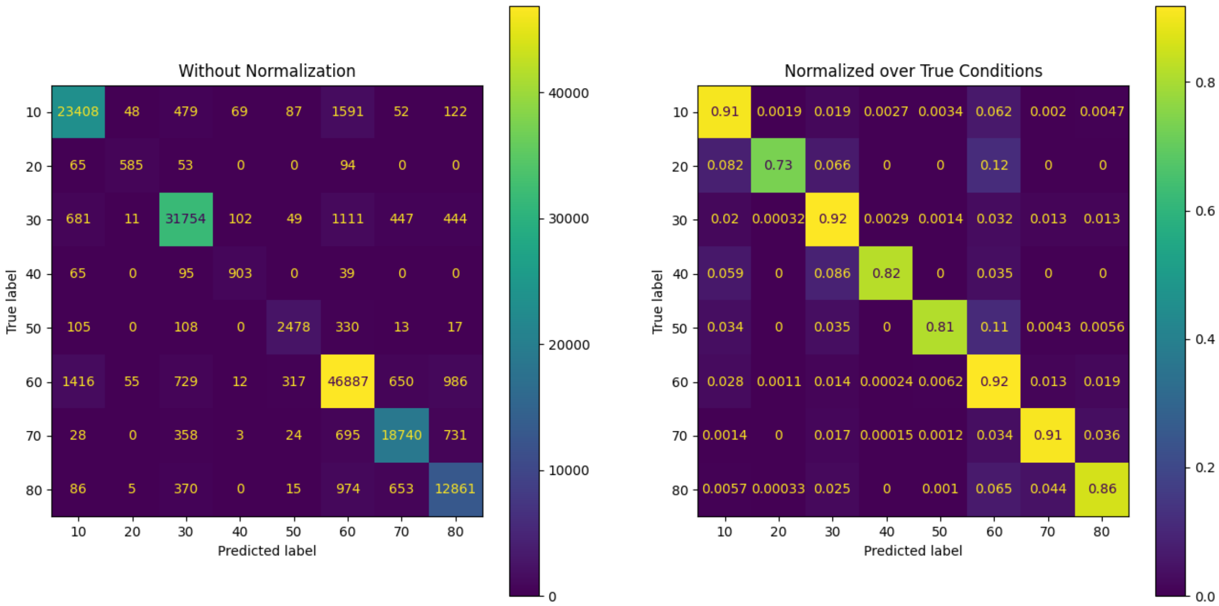

Figure 61: Confusion Matrix of FR_silver of `jobad_bert_finetune_multi`

Language: IT
- F-score (micro) 0.9049
- F-score (macro) 0.8651
- Accuracy 0.9049

By class:

	precision	recall	f1-score	support
60	0.9029	0.9092	0.9060	45968
30	0.9368	0.9254	0.9311	33113
10	0.8990	0.9035	0.9013	24143
70	0.9245	0.9193	0.9219	17926
80	0.8456	0.8673	0.8563	13569
50	0.8317	0.7732	0.8013	2786
40	0.7965	0.7862	0.7913	1085
20	0.8630	0.7657	0.8115	683
accuracy			0.9049	139273
macro avg	0.8750	0.8562	0.8651	139273
weighted avg	0.9050	0.9049	0.9049	139273

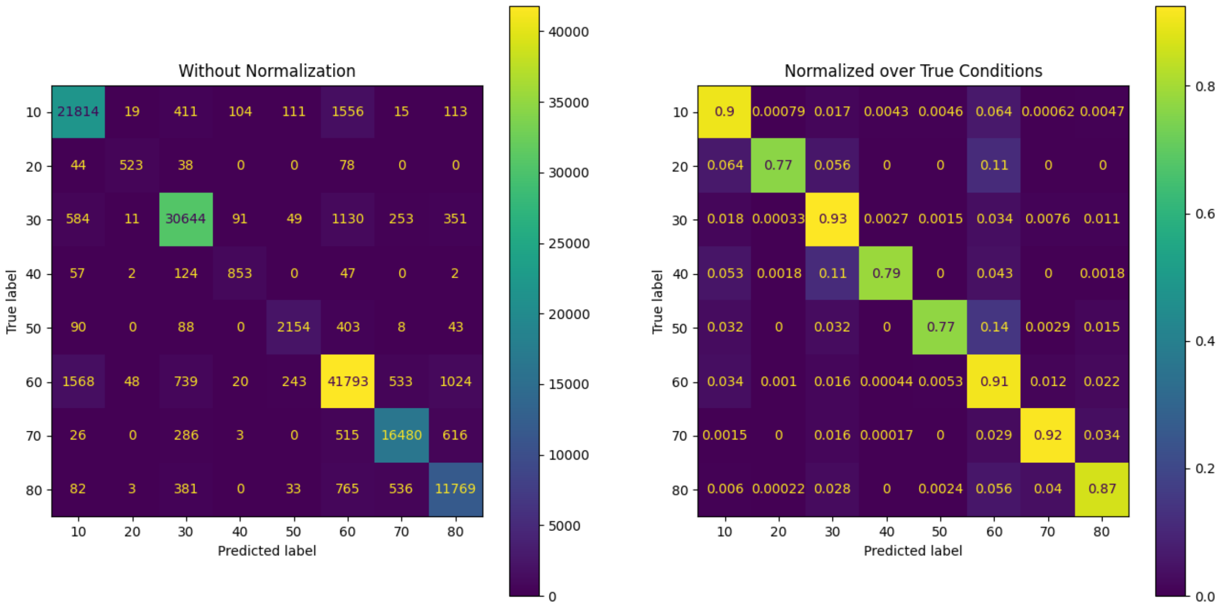

Figure 62: Confusion Matrix of IT_silver of `jobad_bert_finetune_multi`

```
Test set: gold
Language: EN-US
- F-score (micro) 0.8962
- F-score (macro) 0.6995
- Accuracy 0.8962

By class:
```

	precision	recall	f1-score	support
60	0.9073	0.9711	0.9381	3597
10	0.9440	0.8054	0.8692	1506
30	0.9181	0.8267	0.8700	854
70	0.9241	0.8993	0.9116	745
80	0.8283	0.8372	0.8327	559
50	0.4694	0.3286	0.3866	70
40	0.0316	0.7500	0.0606	4
20	1.0000	0.5714	0.7273	7
accuracy			0.8962	7342
macro avg	0.7528	0.7487	0.6995	7342
weighted avg	0.9072	0.8962	0.8994	7342

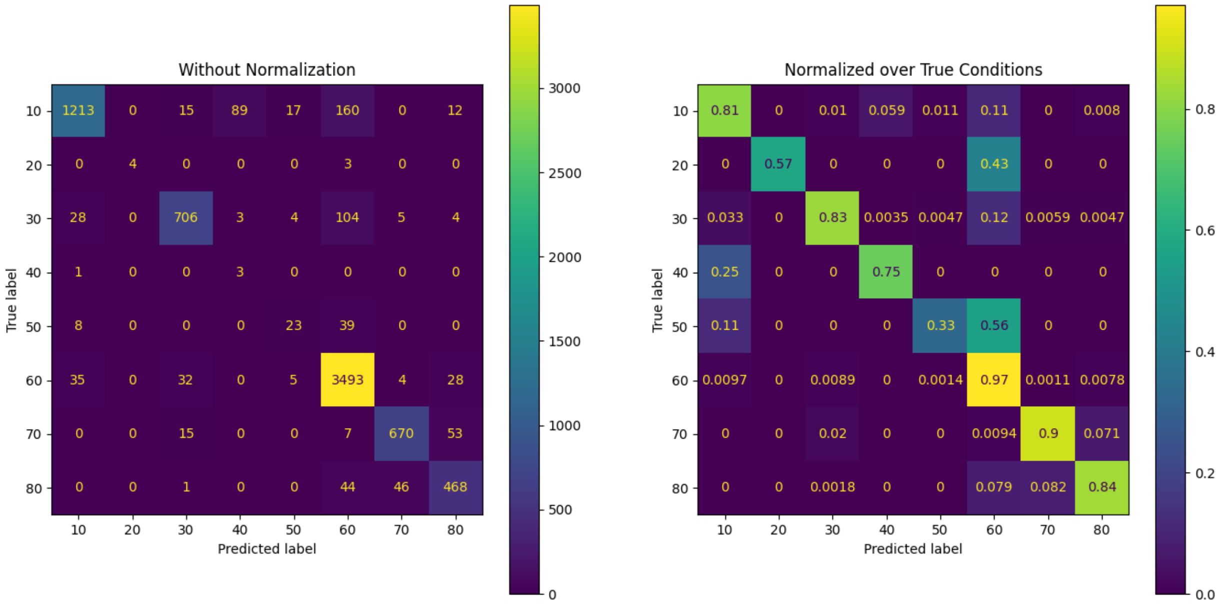

Figure 63: Confusion Matrix of EN-US_gold of `jobad_bert_finetune_multi`

```
Test set: gold
Language: FR
- F-score (micro) 0.8837
- F-score (macro) 0.8225
- Accuracy 0.8837

By class:
              precision    recall  f1-score   support

          60     0.8533    0.9240    0.8872      2499
          30     0.9555    0.8838    0.9183      1653
          70     0.9074    0.8569    0.8814      1223
          10     0.8842    0.8875    0.8858      1058
          80     0.8709    0.8236    0.8466       958
          50     0.7632    0.9355    0.8406       124
          40     0.7273    0.6897    0.7080        58
          20     0.5660    0.6667    0.6122        45

    accuracy                         0.8837      7618
   macro avg     0.8160    0.8335    0.8225      7618
weighted avg     0.8865    0.8837    0.8840      7618
```

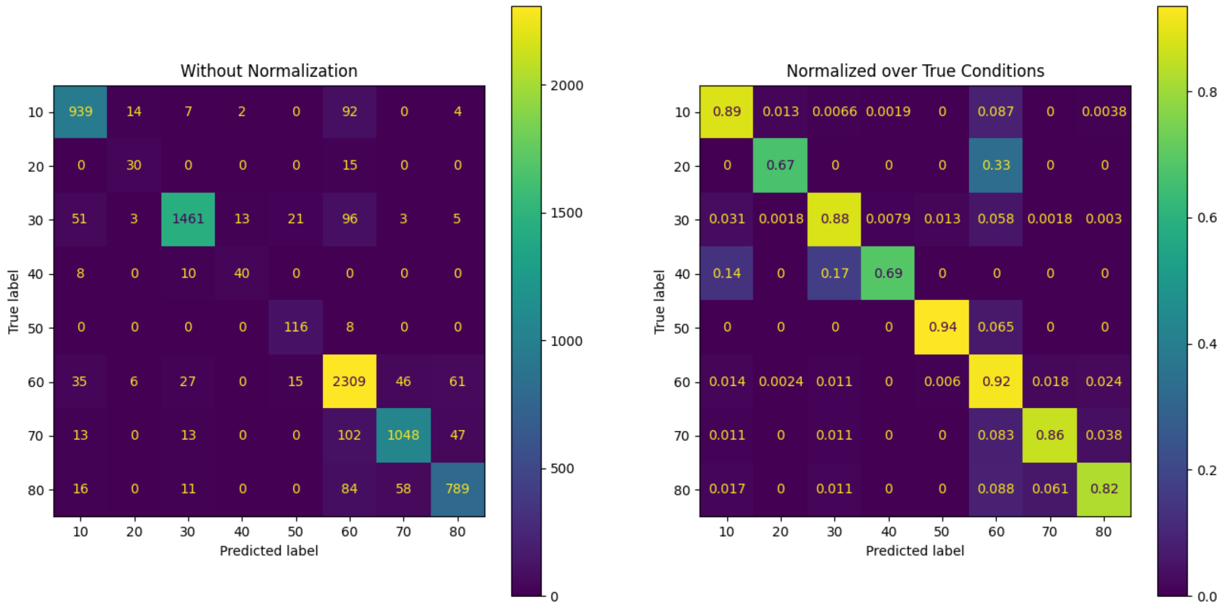

Figure 64: Confusion Matrix of FR_gold of `jobad_bert_finetune_multi`

```
Test set: gold
Language: IT
- F-score (micro) 0.8717
- F-score (macro) 0.8057
- Accuracy 0.8717

By class:
              precision    recall  f1-score   support

          30     0.9437    0.8780    0.9096      2442
          60     0.8244    0.8816    0.8520      2077
          10     0.8933    0.9110    0.9021      1461
          70     0.9520    0.8793    0.9142       903
          80     0.7415    0.7622    0.7517       715
          50     0.7275    0.8215    0.7717       325
          20     0.5667    1.0000    0.7234        17
          40     0.9000    0.4737    0.6207        19

    accuracy                         0.8717      7959
   macro avg     0.8186    0.8259    0.8057      7959
weighted avg     0.8763    0.8717    0.8728      7959
```

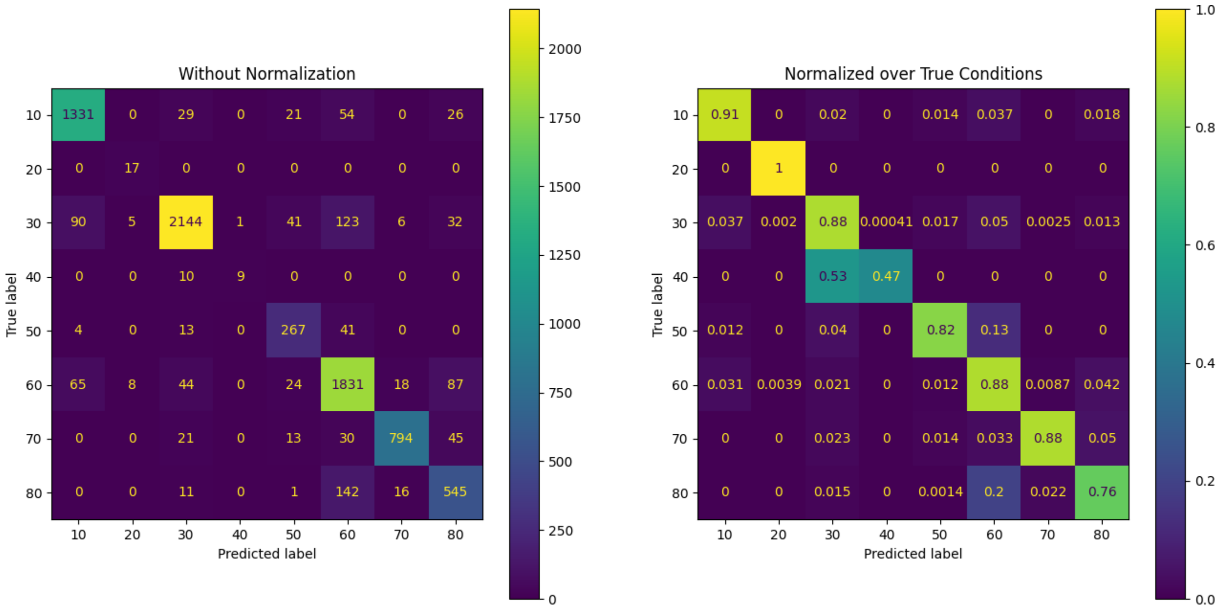

Figure 65: Confusion Matrix of IT_gold of `jobad_bert_finetune_multi`

```
Test set: gold
Language: DE
- F-score (micro) 0.9194
- F-score (macro) 0.8778
- Accuracy 0.9194

By class:
              precision    recall  f1-score   support

          60     0.9223    0.9332    0.9277     38275
          30     0.9415    0.9241    0.9327     30976
          10     0.9196    0.9190    0.9193     20636
          70     0.9286    0.9268    0.9277     15307
          80     0.8706    0.8747    0.8726     11003
          50     0.8313    0.8457    0.8385      1964
          40     0.7696    0.8496    0.8076       924
          20     0.8061    0.7862    0.7960       608

    accuracy                         0.9194    119693
   macro avg     0.8737    0.8824    0.8778    119693
weighted avg     0.9196    0.9194    0.9194    119693
```

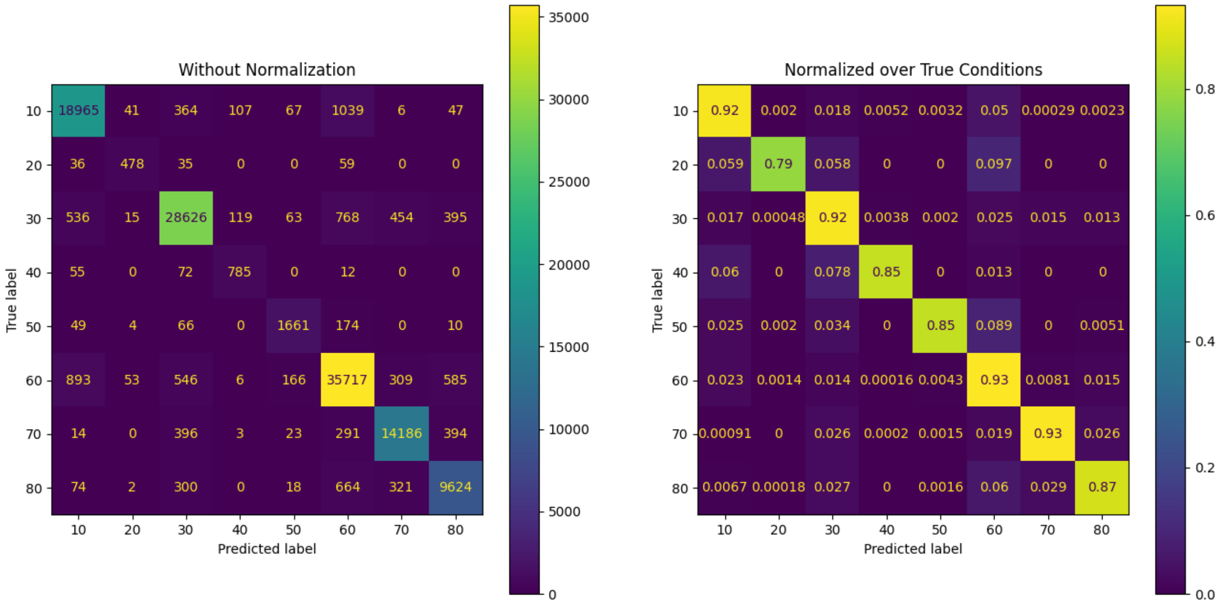

Figure 66: Confusion Matrix of DE_gold of `jobad_bert_finetune_multi`

C.8 `xlm-roberta-base_w_context`

```
Test set: silver
Language: EN-US
- F-score (micro) 0.9235
- F-score (macro) 0.8805
- Accuracy 0.9235

By class:
              precision    recall  f1-score   support

          60     0.9328    0.9248    0.9287     44085
          30     0.9414    0.9332    0.9373     33472
          10     0.9050    0.9233    0.9140     23052
          70     0.9369    0.9392    0.9380     17932
          80     0.8924    0.9062    0.8993     12569
          50     0.8588    0.8501    0.8544      2589
          40     0.7561    0.8080    0.7811      1005
          20     0.8333    0.7526    0.7909       764

    accuracy                         0.9235    135468
```

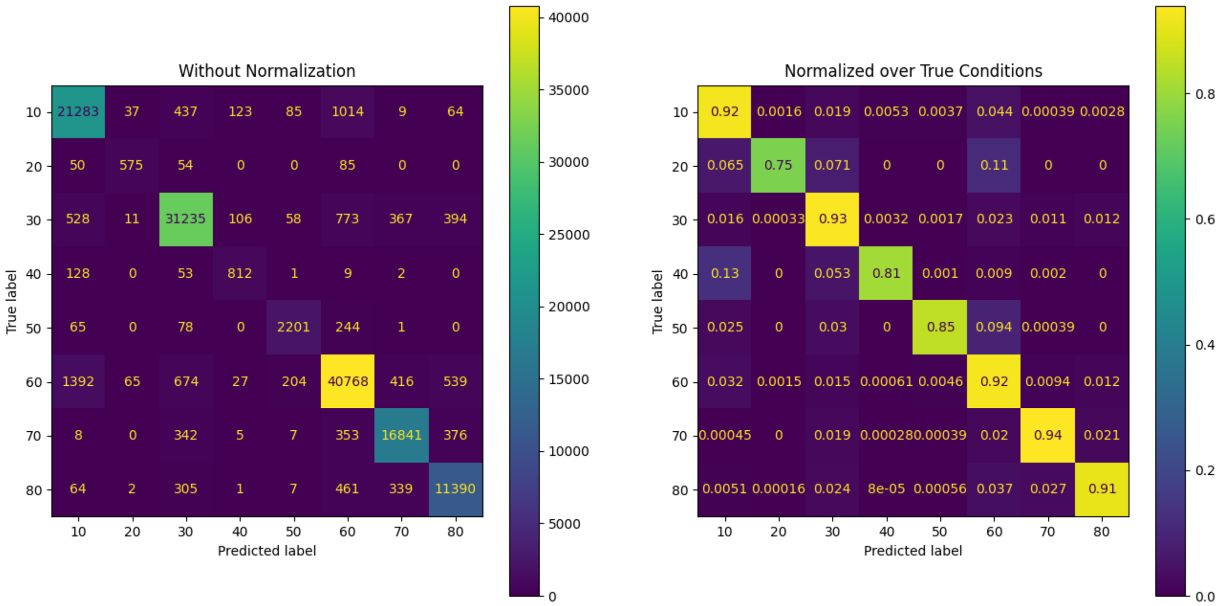

Figure 67: Confusion Matrix of EN-US_silver of `xlm-roberta-base_w_context`

```
   macro avg      0.8821     0.8797     0.8805     135468
weighted avg      0.9237     0.9235     0.9235     135468
```

```
Test set: silver
Language: FR
- F-score (micro) 0.9308
- F-score (macro) 0.89
- Accuracy 0.9308

By class:
              precision    recall  f1-score   support

          60     0.9403    0.9326    0.9364     51052
          30     0.9430    0.9408    0.9419     34599
          10     0.9132    0.9304    0.9217     25856
          70     0.9452    0.9445    0.9448     20579
          80     0.9061    0.9150    0.9105     14964
          50     0.8846    0.8745    0.8795      3051
          40     0.7671    0.7623    0.7647      1102
          20     0.8639    0.7804    0.8200       797
```

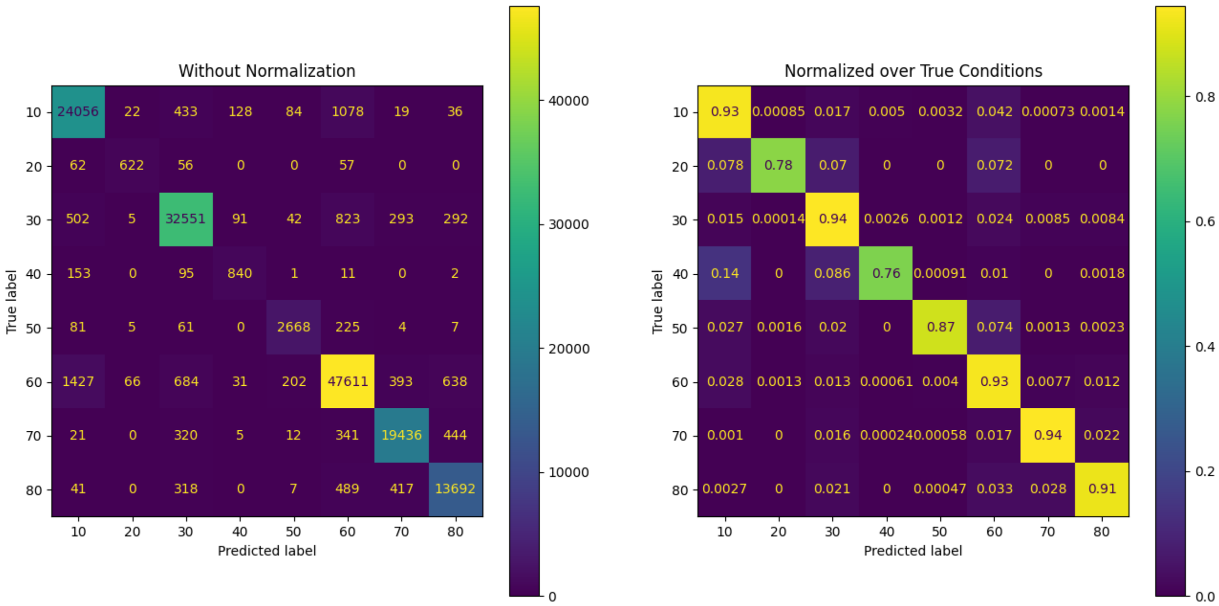

Figure 68: Confusion Matrix of FR_silver of `xlm-roberta-base_w_context`

```
    accuracy                         0.9308    152000
   macro avg     0.8954    0.8850    0.8900    152000
weighted avg     0.9308    0.9308    0.9308    152000


Test set: silver
Language: IT
- F-score (micro) 0.9291
- F-score (macro) 0.8908
- Accuracy 0.9291

By class:
              precision    recall  f1-score   support

          60     0.9339    0.9289    0.9314     45968
          30     0.9493    0.9430    0.9461     33113
          10     0.9051    0.9258    0.9153     24143
          70     0.9455    0.9447    0.9451     17926
          80     0.9135    0.9142    0.9138     13569
          50     0.8582    0.8557    0.8569      2786
          40     0.8110    0.8028    0.8069      1085
          20     0.8567    0.7701    0.8111       683
```

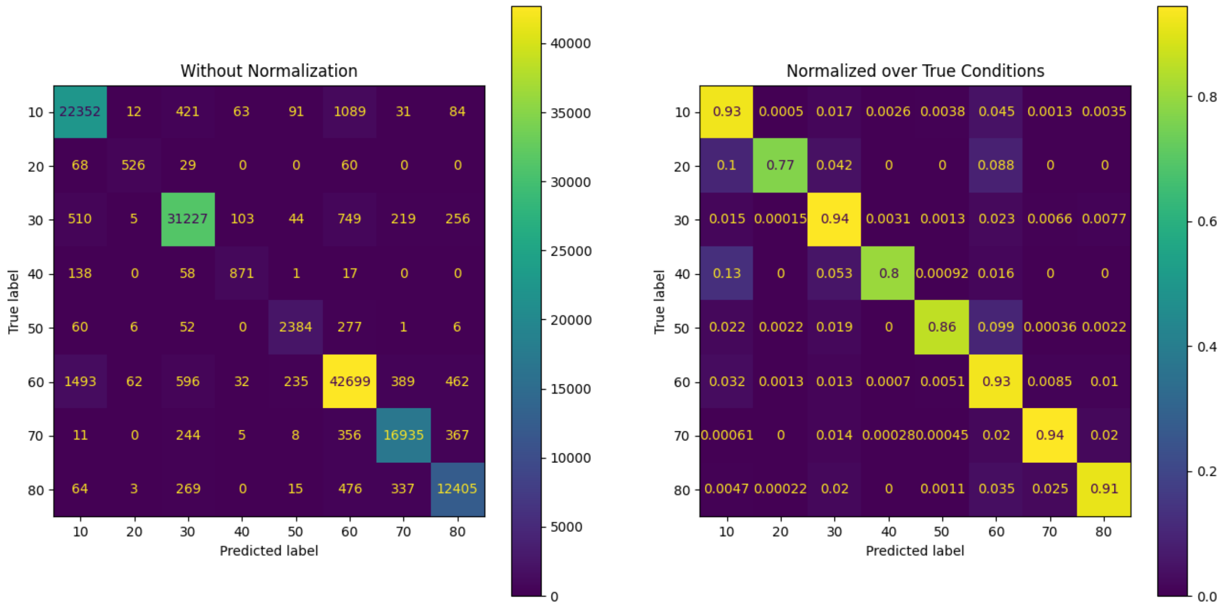

Figure 69: Confusion Matrix of IT_silver of `xlm-roberta-base_w_context`

```
    accuracy                           0.9291    139273
   macro avg     0.8966    0.8857    0.8908    139273
weighted avg     0.9292    0.9291    0.9291    139273


Test set: gold
Language: EN-US
- F-score (micro) 0.9226
- F-score (macro) 0.7665
- Accuracy 0.9226

By class:
              precision    recall  f1-score   support

          60     0.9493    0.9536    0.9515      3597
          10     0.9353    0.8738    0.9035      1506
          30     0.9399    0.8970    0.9179       854
          70     0.9358    0.9584    0.9469       745
          80     0.8481    0.8587    0.8533       559
          50     0.7619    0.9143    0.8312        70
          40     0.0000    0.0000    0.0000         4
```

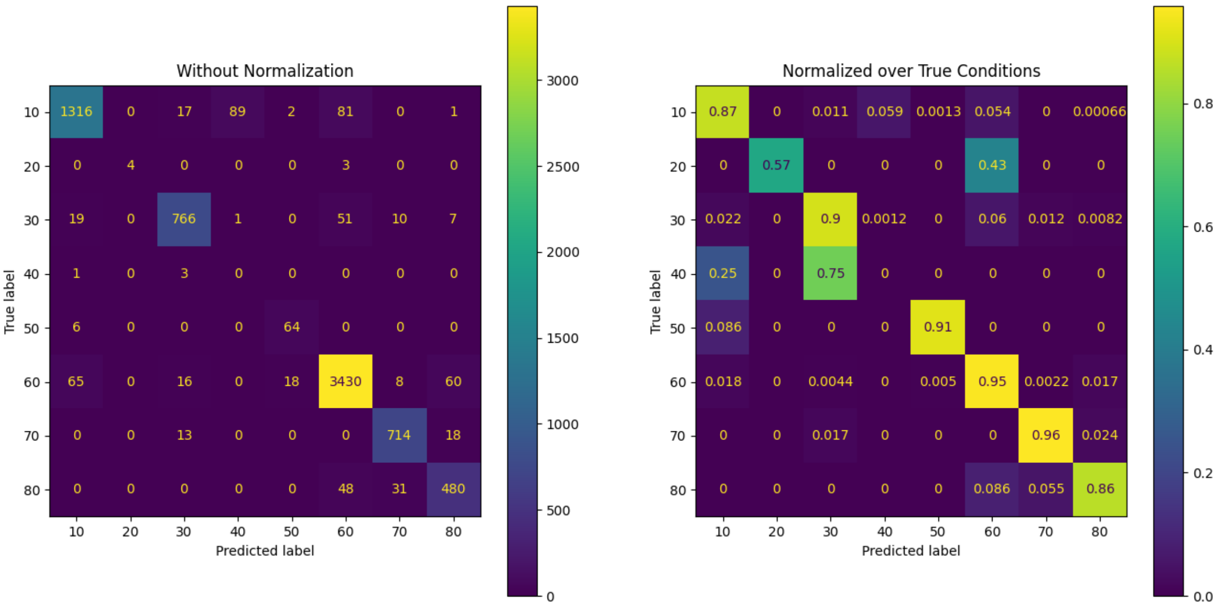

Figure 70: Confusion Matrix of EN-US_gold of `xlm-roberta-base_w_context`

20	1.0000	0.5714	0.7273	7
accuracy			0.9226	7342
macro avg	0.7963	0.7534	0.7665	7342
weighted avg	0.9340	0.9226	0.9279	7342

```
Test set: gold
Language: FR
- F-score (micro) 0.9319
- F-score (macro) 0.8664
- Accuracy 0.9319
```

By class:

	precision	recall	f1-score	support
60	0.9326	0.9360	0.9343	2499
30	0.9481	0.9286	0.9383	1653
70	0.9621	0.9550	0.9586	1223
10	0.9110	0.9575	0.9336	1058
80	0.9115	0.9134	0.9124	958
50	0.8254	0.8387	0.8320	124

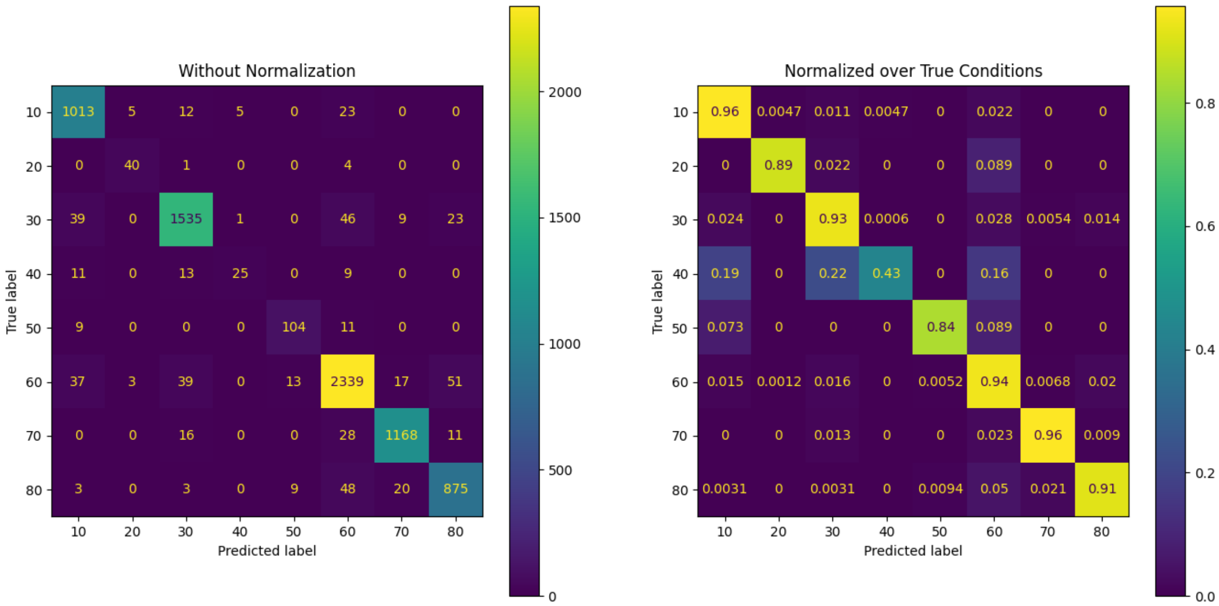

Figure 71: Confusion Matrix of FR_gold of `xlm-roberta-base_w_context`

20	0.8333	0.8889	0.8602	45
40	0.8065	0.4310	0.5618	58
accuracy			0.9319	7618
macro avg	0.8913	0.8561	0.8664	7618
weighted avg	0.9318	0.9319	0.9313	7618

```
Test set: gold
Language: IT
- F-score (micro) 0.908
- F-score (macro) 0.8579
- Accuracy 0.908
```

By class:

	precision	recall	f1-score	support
30	0.9351	0.9140	0.9244	2442
60	0.9146	0.8767	0.8953	2077
10	0.8904	0.9452	0.9170	1461
70	0.9635	0.9347	0.9488	903
80	0.8044	0.9147	0.8560	715

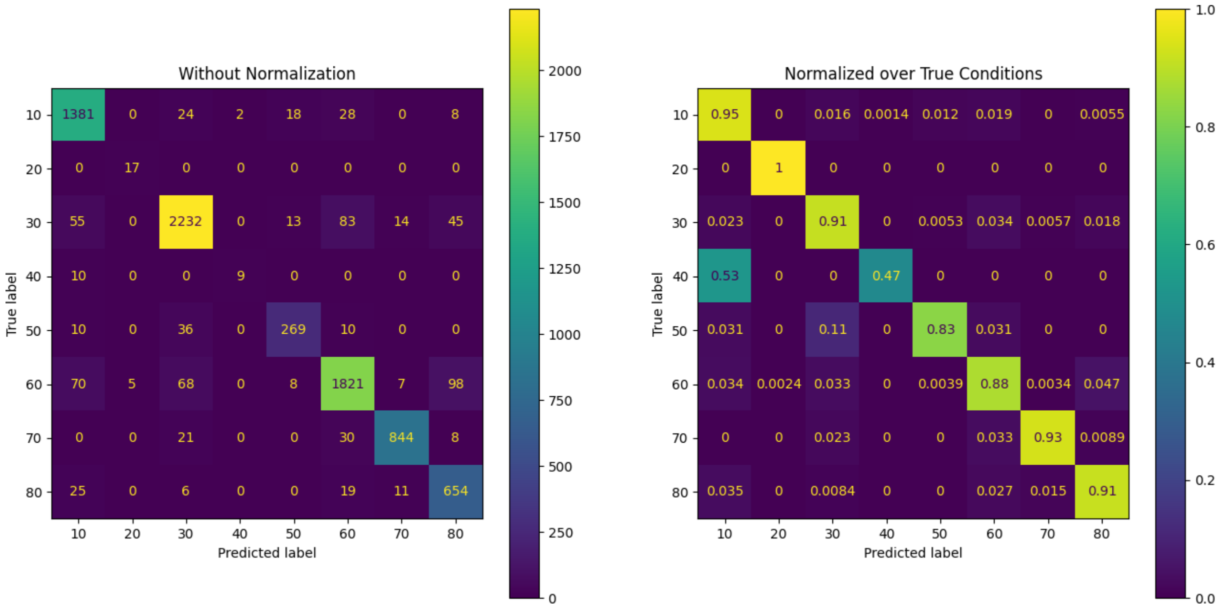

Figure 72: Confusion Matrix of IT_gold of `xlm-roberta-base_w_context`

	precision	recall	f1-score	support
50	0.8734	0.8277	0.8499	325
20	0.7727	1.0000	0.8718	17
40	0.8182	0.4737	0.6000	19
accuracy			0.9080	7959
macro avg	0.8715	0.8608	0.8579	7959
weighted avg	0.9099	0.9080	0.9081	7959

```
Test set: gold
Language: DE
- F-score (micro) 0.917
- F-score (macro) 0.8727
- Accuracy 0.917

By class:
```

	precision	recall	f1-score	support
60	0.9261	0.9204	0.9232	38275
30	0.9381	0.9240	0.9310	30976
10	0.9035	0.9229	0.9131	20636
70	0.9198	0.9393	0.9294	15307

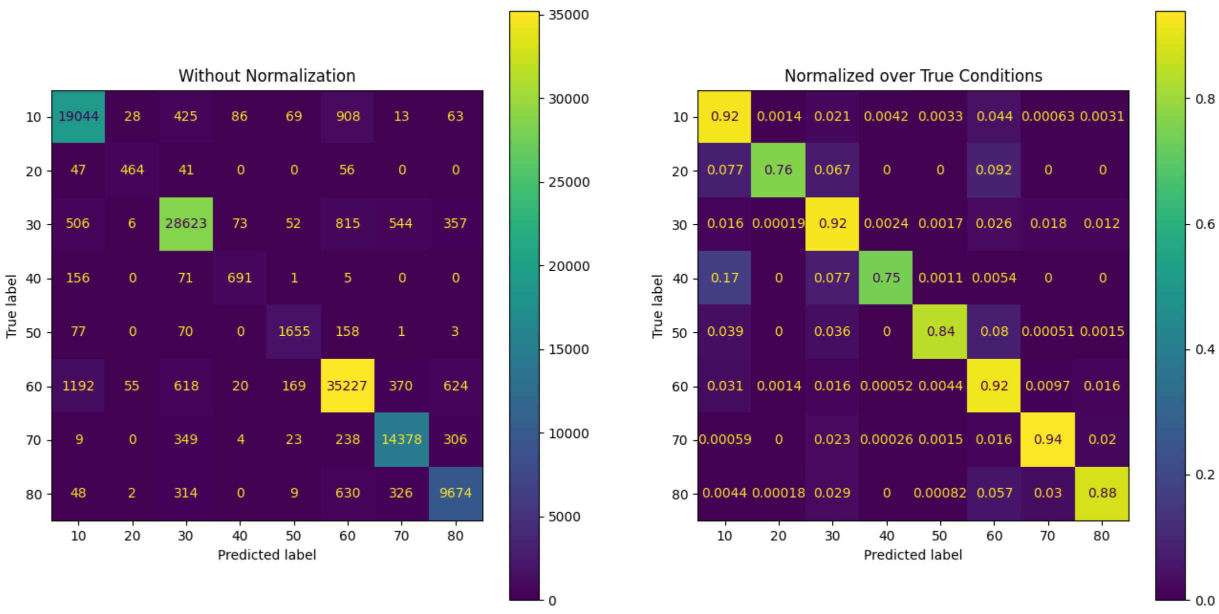

Figure 73: Confusion Matrix of DE_gold of `xlm-roberta-base_w_context`

80	0.8773	0.8792	0.8783	11003
50	0.8367	0.8427	0.8397	1964
40	0.7906	0.7478	0.7686	924
20	0.8360	0.7632	0.7979	608
accuracy			0.9170	119693
macro avg	0.8785	0.8674	0.8727	119693
weighted avg	0.9171	0.9170	0.9170	119693

C.9 `xlm-roberta-base_o_context`

```
Test set: silver
Language: EN-US
- F-score (micro) 0.9178
- F-score (macro) 0.8724
- Accuracy 0.9178

By class:
                precision    recall  f1-score   support
```

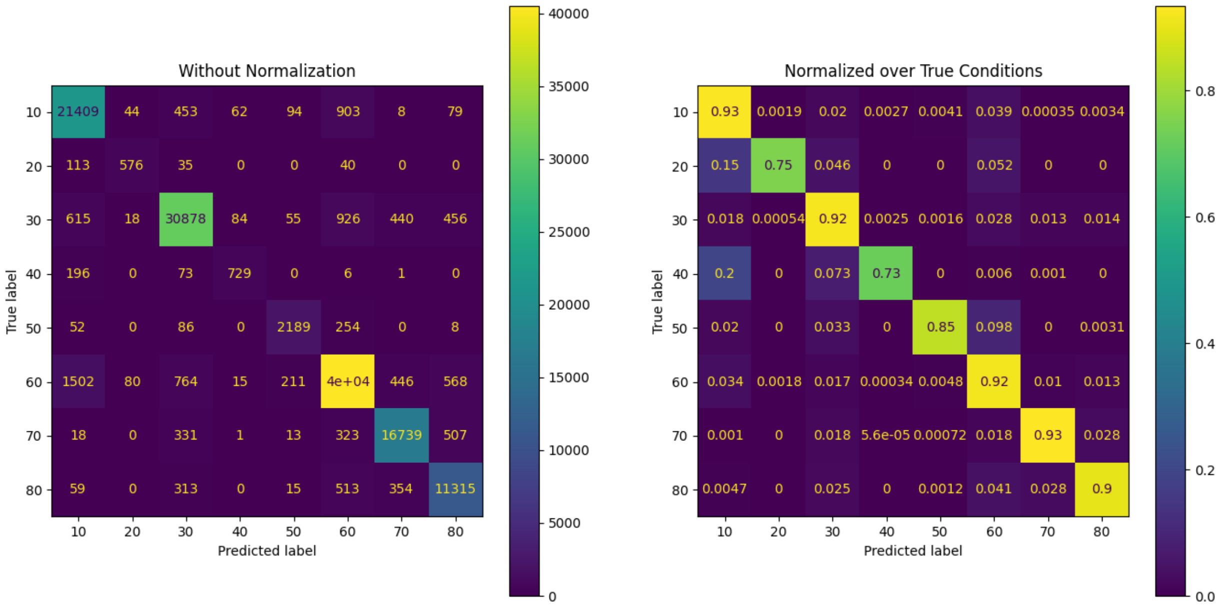

Figure 74: Confusion Matrix of EN-US_silver of `xlm-roberta-base_o_context`

	precision	recall	f1-score	support
60	0.9318	0.9187	0.9252	44085
30	0.9376	0.9225	0.9300	33472
10	0.8934	0.9287	0.9107	23052
70	0.9306	0.9335	0.9320	17932
80	0.8749	0.9002	0.8874	12569
50	0.8494	0.8455	0.8475	2589
40	0.8182	0.7254	0.7690	1005
20	0.8022	0.7539	0.7773	764
accuracy			0.9178	135468
macro avg	0.8798	0.8660	0.8724	135468
weighted avg	0.9181	0.9178	0.9178	135468

```
Test set: silver
Language: FR
- F-score (micro) 0.9236
- F-score (macro) 0.8779
- Accuracy 0.9236

By class:
              precision    recall  f1-score   support
```

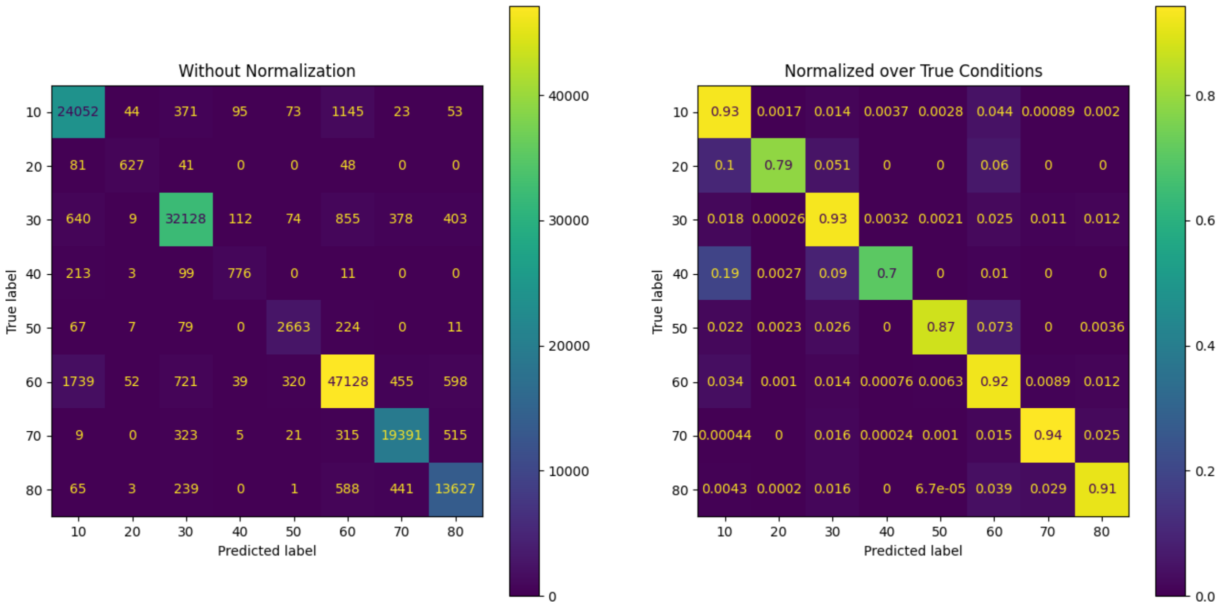

Figure 75: Confusion Matrix of FR_silver of `xlm-roberta-base_o_context`

60	0.9367	0.9231	0.9299	51052
30	0.9449	0.9286	0.9367	34599
10	0.8953	0.9302	0.9124	25856
70	0.9373	0.9423	0.9398	20579
80	0.8961	0.9107	0.9033	14964
50	0.8449	0.8728	0.8586	3051
40	0.7556	0.7042	0.7290	1102
20	0.8416	0.7867	0.8132	797
accuracy			0.9236	152000
macro avg	0.8815	0.8748	0.8779	152000
weighted avg	0.9239	0.9236	0.9237	152000

```
Test set: silver
Language: IT
- F-score (micro) 0.9225
- F-score (macro) 0.8771
- Accuracy 0.9225

By class:
```

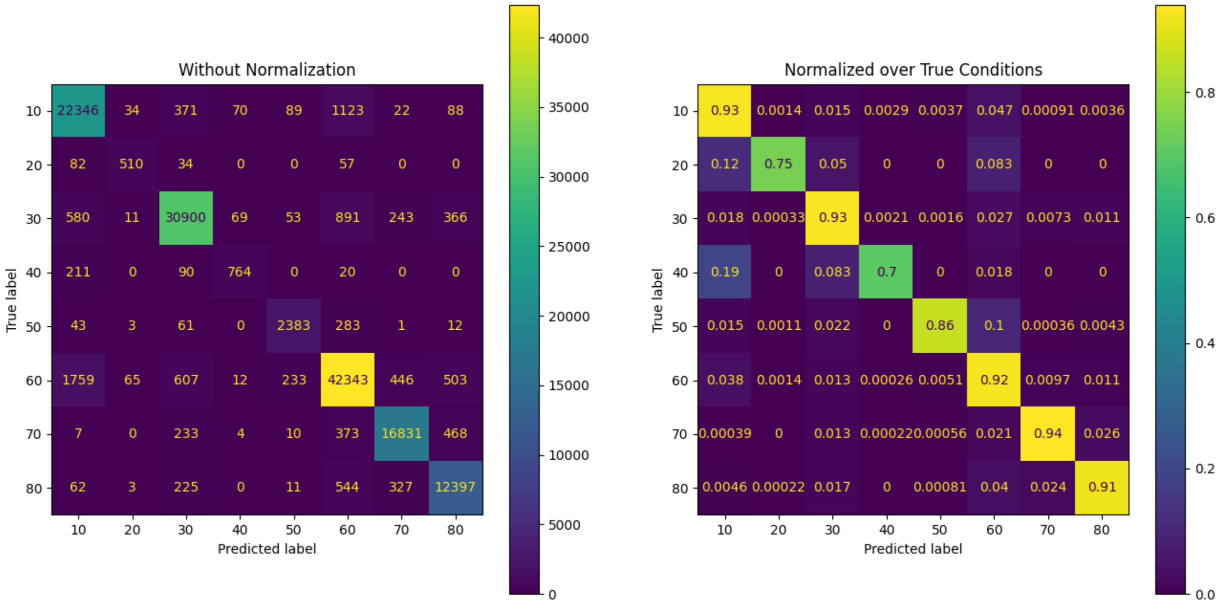

Figure 76: Confusion Matrix of IT_silver of `xlm-roberta-base_o_context`

```
              precision    recall  f1-score   support

          60     0.9279    0.9211    0.9245     45968
          30     0.9502    0.9332    0.9416     33113
          10     0.8906    0.9256    0.9078     24143
          70     0.9419    0.9389    0.9404     17926
          80     0.8961    0.9136    0.9048     13569
          50     0.8575    0.8553    0.8564      2786
          40     0.8313    0.7041    0.7625      1085
          20     0.8147    0.7467    0.7792       683

    accuracy                         0.9225    139273
   macro avg     0.8888    0.8673    0.8771    139273
weighted avg     0.9227    0.9225    0.9224    139273
```

```
Test set: gold
Language: EN-US
- F-score (micro) 0.9254
- F-score (macro) 0.764
- Accuracy 0.9254
```

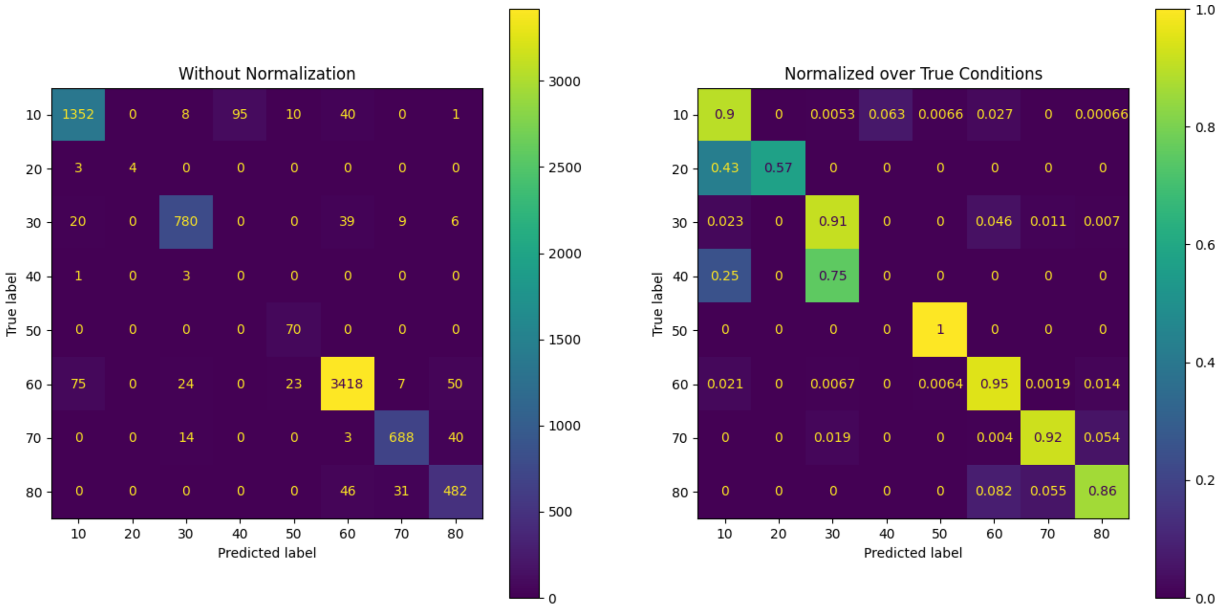

Figure 77: Confusion Matrix of EN-US_gold of `xlm-roberta-base_o_context`

```
By class:
              precision    recall  f1-score   support

          60     0.9639    0.9502    0.9570      3597
          10     0.9318    0.8977    0.9144      1506
          30     0.9409    0.9133    0.9269       854
          70     0.9361    0.9235    0.9297       745
          80     0.8325    0.8623    0.8471       559
          50     0.6796    1.0000    0.8092        70
          40     0.0000    0.0000    0.0000         4
          20     1.0000    0.5714    0.7273         7

    accuracy                         0.9254      7342
   macro avg     0.7856    0.7648    0.7640      7342
weighted avg     0.9386    0.9254    0.9315      7342
```

```
Test set: gold
Language: FR
- F-score (micro) 0.918
- F-score (macro) 0.862
- Accuracy 0.918
```

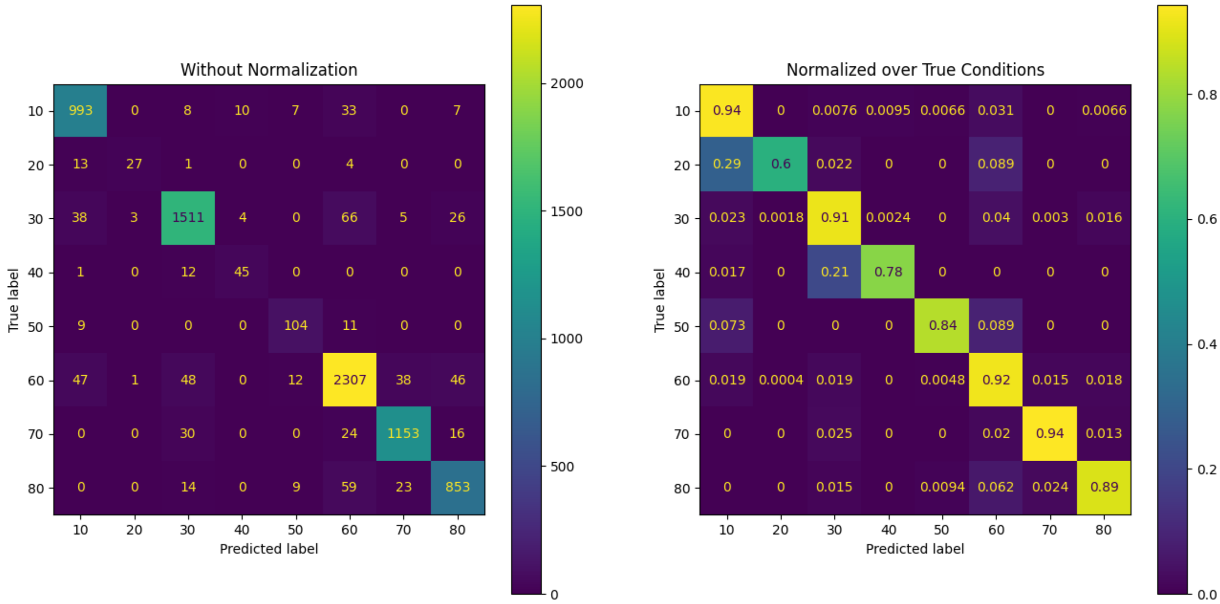

Figure 78: Confusion Matrix of FR_gold of `xlm-roberta-base_o_context`

```
By class:
              precision    recall  f1-score   support

          60     0.9213    0.9232    0.9222      2499
          30     0.9304    0.9141    0.9222      1653
          70     0.9459    0.9428    0.9443      1223
          10     0.9019    0.9386    0.9199      1058
          80     0.8998    0.8904    0.8951       958
          50     0.7879    0.8387    0.8125       124
          40     0.7627    0.7759    0.7692        58
          20     0.8710    0.6000    0.7105        45

    accuracy                         0.9180      7618
   macro avg     0.8776    0.8529    0.8620      7618
weighted avg     0.9182    0.9180    0.9178      7618
```

```
Test set: gold
Language: IT
- F-score (micro) 0.9034
- F-score (macro) 0.848
```

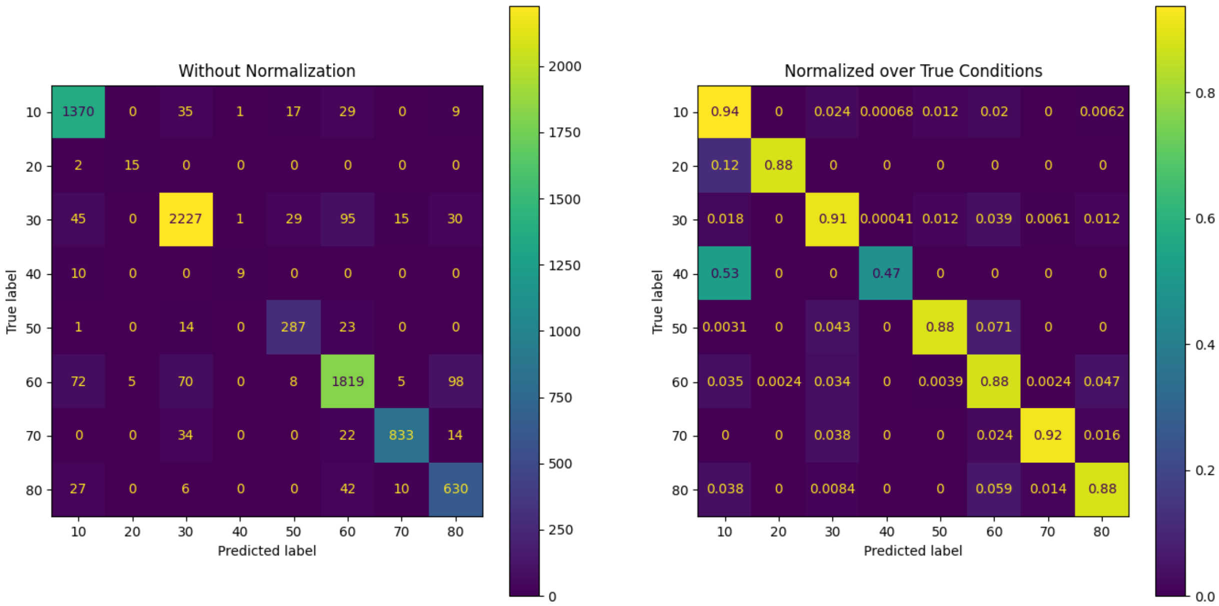

Figure 79: Confusion Matrix of IT_gold of `xlm-roberta-base_o_context`

```
- Accuracy 0.9034

By class:
              precision    recall  f1-score   support

          30     0.9334    0.9120    0.9225      2442
          60     0.8961    0.8758    0.8858      2077
          10     0.8972    0.9377    0.9170      1461
          70     0.9652    0.9225    0.9434       903
          80     0.8067    0.8811    0.8422       715
          50     0.8416    0.8831    0.8619       325
          20     0.7500    0.8824    0.8108        17
          40     0.8182    0.4737    0.6000        19

    accuracy                         0.9034      7959
   macro avg     0.8635    0.8460    0.8480      7959
weighted avg     0.9048    0.9034    0.9036      7959

Test set: gold
Language: DE
- F-score (micro) 0.9089
```

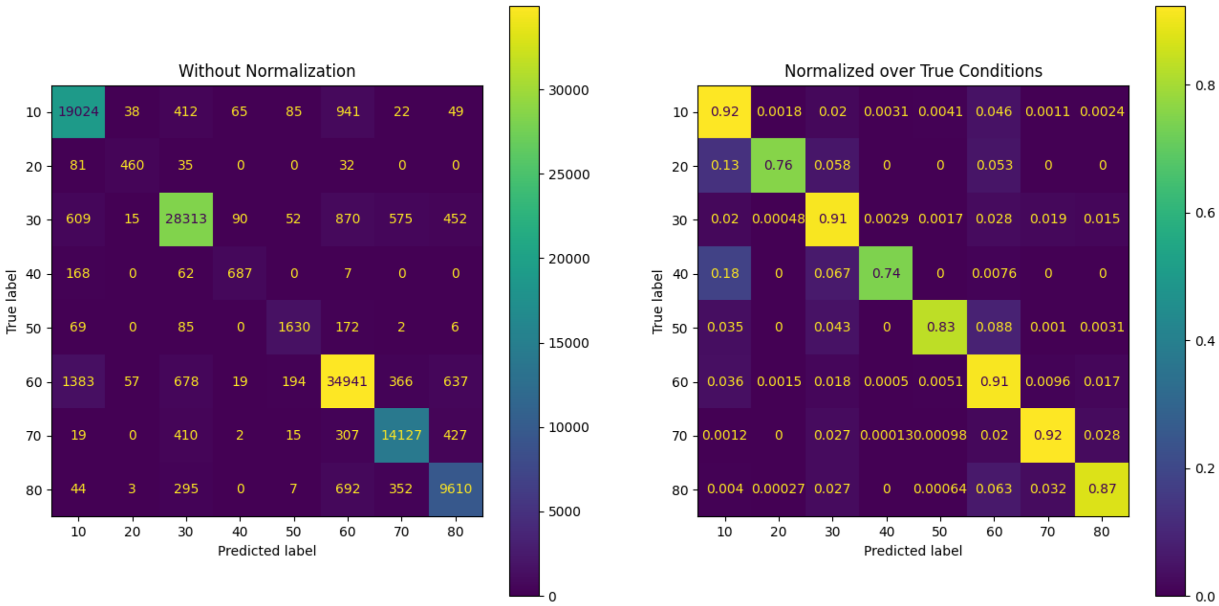

Figure 80: Confusion Matrix of DE_gold of `xlm-roberta-base_o_context`

```
- F-score (macro) 0.8631
- Accuracy 0.9089

By class:
              precision    recall  f1-score   support

          60     0.9204    0.9129    0.9166     38275
          30     0.9347    0.9140    0.9243     30976
          10     0.8891    0.9219    0.9052     20636
          70     0.9147    0.9229    0.9188     15307
          80     0.8595    0.8734    0.8664     11003
          50     0.8220    0.8299    0.8259      1964
          40     0.7961    0.7435    0.7689       924
          20     0.8028    0.7566    0.7790       608

    accuracy                         0.9089    119693
   macro avg     0.8674    0.8594    0.8631    119693
weighted avg     0.9092    0.9089    0.9090    119693
```

C.10 xlm-roberta-base_w_context_2_DE_sents

Test set: gold

Language: DE

- F-score (micro) 0.9154
- F-score (macro) 0.8729
- Accuracy 0.9154

By class:

	precision	recall	f1-score	support
60	0.9262	0.9153	0.9207	38275
30	0.9371	0.9234	0.9302	30976
10	0.8978	0.9243	0.9108	20636
70	0.9201	0.9371	0.9285	15307
80	0.8740	0.8803	0.8771	11003
50	0.8242	0.8473	0.8356	1964
40	0.8141	0.7348	0.7725	924
20	0.8287	0.7878	0.8078	608
accuracy			0.9154	119693
macro avg	0.8778	0.8688	0.8729	119693
weighted avg	0.9155	0.9154	0.9153	119693

C.11 xlm-roberta-base_w_context_2_EN-US_sents

Test set: silver

Language: EN-US

- F-score (micro) 0.9235
- F-score (macro) 0.8853
- Accuracy 0.9235

By class:

	precision	recall	f1-score	support
60	0.9345	0.9226	0.9285	44085
30	0.9372	0.9365	0.9368	33472

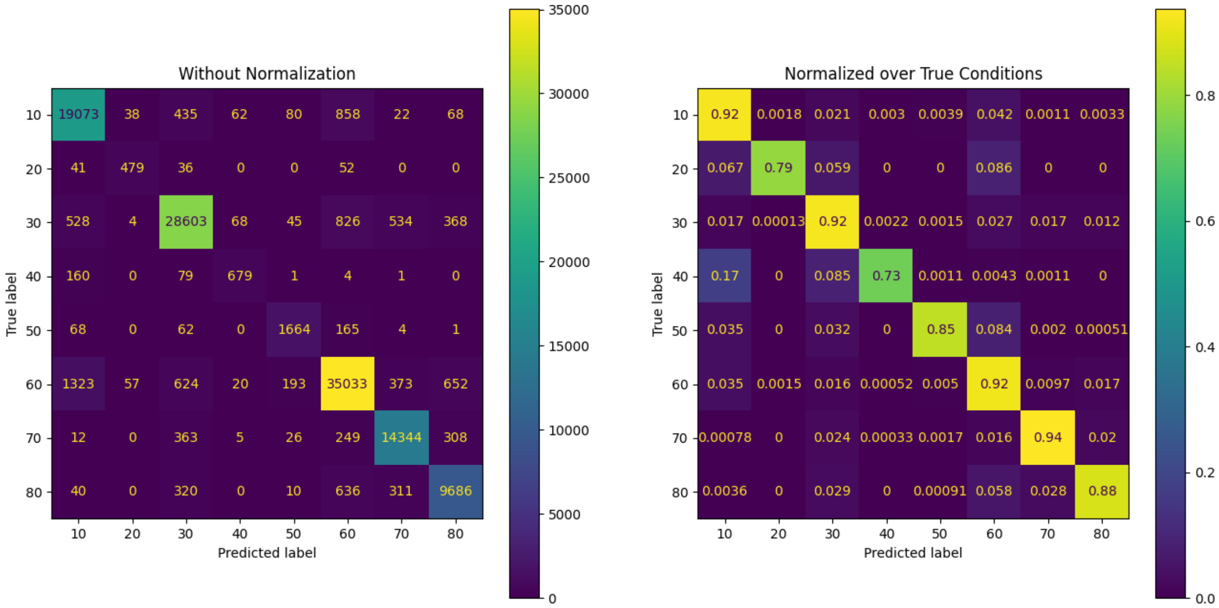

Figure 81: Confusion Matrix of DE_gold of `xlm-roberta-base_w_context_2_DE_sents`

	precision	recall	f1-score	support
10	0.9008	0.9272	0.9138	23052
70	0.9386	0.9373	0.9379	17932
80	0.8956	0.9002	0.8979	12569
50	0.8751	0.8443	0.8594	2589
40	0.8036	0.7980	0.8008	1005
20	0.8281	0.7880	0.8075	764
accuracy			0.9235	135468
macro avg	0.8892	0.8818	0.8853	135468
weighted avg	0.9236	0.9235	0.9235	135468

```
Test set: gold
Language: EN-US
- F-score (micro) 0.9151
- F-score (macro) 0.7568
- Accuracy 0.9151
```

By class:

	precision	recall	f1-score	support
60	0.9474	0.9569	0.9521	3597

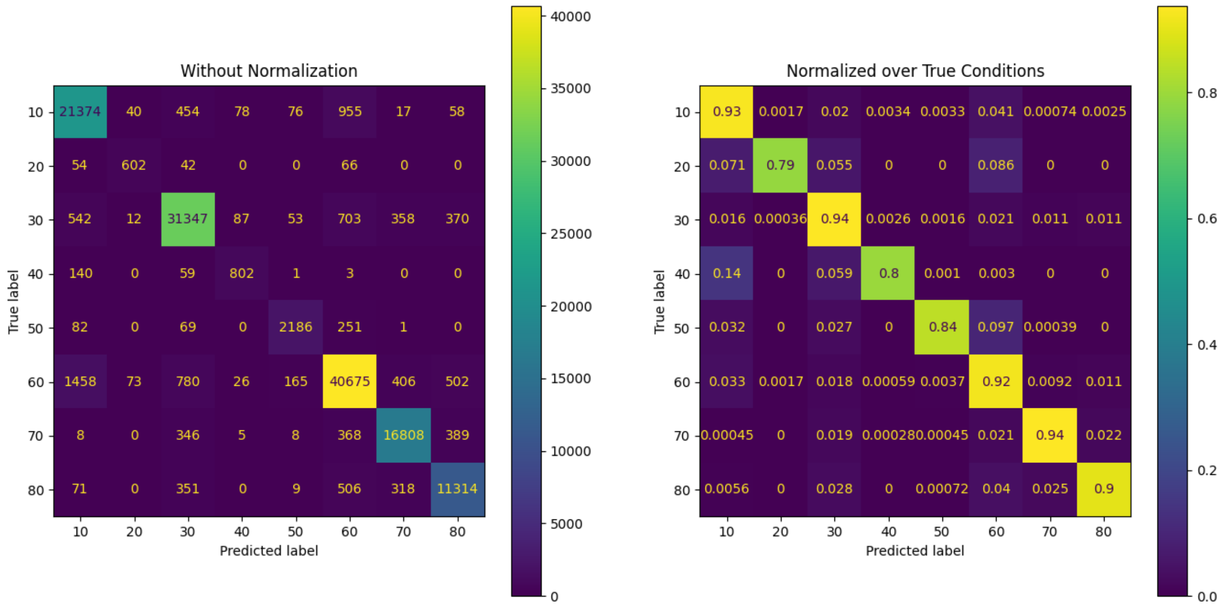

Figure 82: Confusion Matrix of EN-US_silver of xlm-roberta-base_w_context_2_EN-US_sents

10	0.9309	0.8493	0.8882	1506
30	0.9423	0.8993	0.9203	854
70	0.9182	0.9490	0.9333	745
80	0.8606	0.8283	0.8441	559
50	0.7361	0.7571	0.7465	70
40	0.0221	0.7500	0.0429	4
20	1.0000	0.5714	0.7273	7
accuracy			0.9151	7342
macro avg	0.7947	0.8202	0.7568	7342
weighted avg	0.9314	0.9151	0.9225	7342

C.12 xlm-roberta-base_w_context_2_FR_sents

Test set: silver

Language: FR

- F-score (micro) 0.9308
- F-score (macro) 0.8892
- Accuracy 0.9308

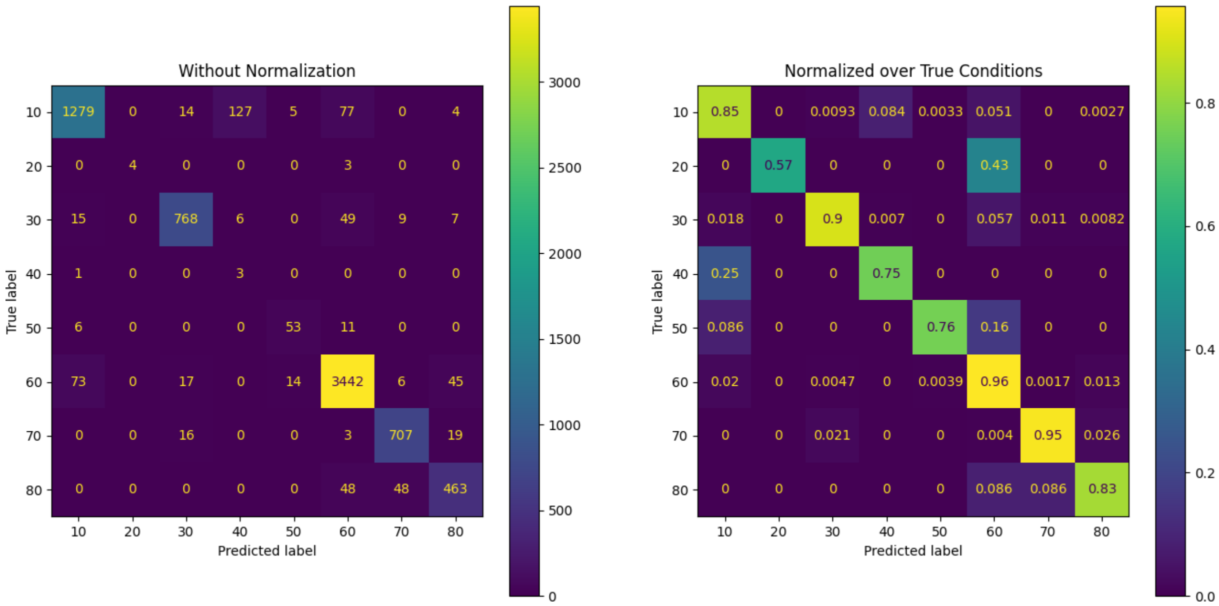

Figure 83: Confusion Matrix of EN-US_gold of `xlm-roberta-base_w_context_2_EN-US_sents`

By class:

	precision	recall	f1-score	support
60	0.9440	0.9310	0.9375	51052
30	0.9411	0.9424	0.9417	34599
10	0.9061	0.9358	0.9207	25856
70	0.9466	0.9399	0.9433	20579
80	0.9057	0.9175	0.9116	14964
50	0.8883	0.8732	0.8807	3051
40	0.7847	0.7241	0.7532	1102
20	0.8908	0.7679	0.8248	797
accuracy			0.9308	152000
macro avg	0.9009	0.8790	0.8892	152000
weighted avg	0.9309	0.9308	0.9308	152000

Test set: gold

Language: FR

- F-score (micro) 0.9303
- F-score (macro) 0.8601

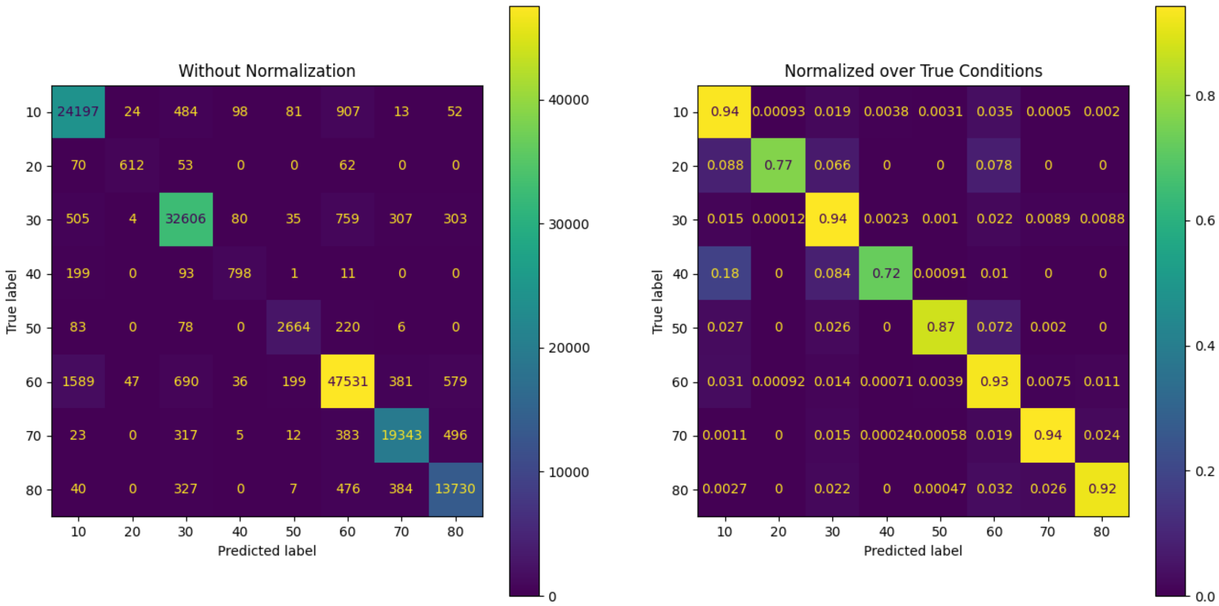

Figure 84: Confusion Matrix of FR_silver of `xlm-roberta-base_w_context_2_FR_sents`

```
- Accuracy 0.9303

By class:
              precision    recall  f1-score   support


          60     0.9323    0.9372    0.9347      2499
          30     0.9510    0.9280    0.9394      1653
          70     0.9704    0.9379    0.9538      1223
          10     0.8983    0.9603    0.9283      1058
          80     0.9064    0.9196    0.9130       958
          50     0.7761    0.8387    0.8062       124
          20     0.8163    0.8889    0.8511        45
          40     0.9200    0.3966    0.5542        58

    accuracy                         0.9303      7618
   macro avg     0.8964    0.8509    0.8601      7618
weighted avg     0.9312    0.9303    0.9297      7618
```

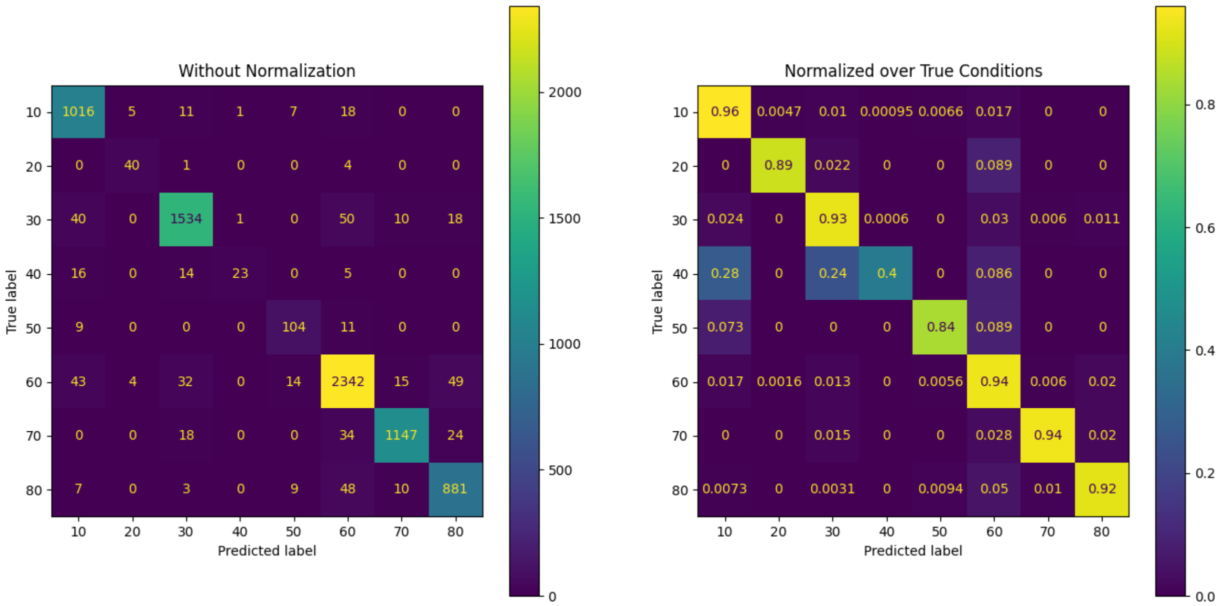

Figure 85: Confusion Matrix of FR_gold of `xlm-roberta-base_w_context_2_FR_sents`

C.13 `xlm-roberta-base_w_context_2_IT_sents`

```
Test set: silver
Language: IT
- F-score (micro) 0.93
- F-score (macro) 0.89
- Accuracy 0.93

By class:
```

	precision	recall	f1-score	support
60	0.9354	0.9280	0.9317	45968
30	0.9495	0.9444	0.9469	33113
10	0.9056	0.9249	0.9152	24143
70	0.9465	0.9463	0.9464	17926
80	0.9144	0.9217	0.9180	13569
50	0.8683	0.8661	0.8672	2786
40	0.8093	0.7825	0.7957	1085
20	0.8220	0.7775	0.7991	683
accuracy			0.9300	139273

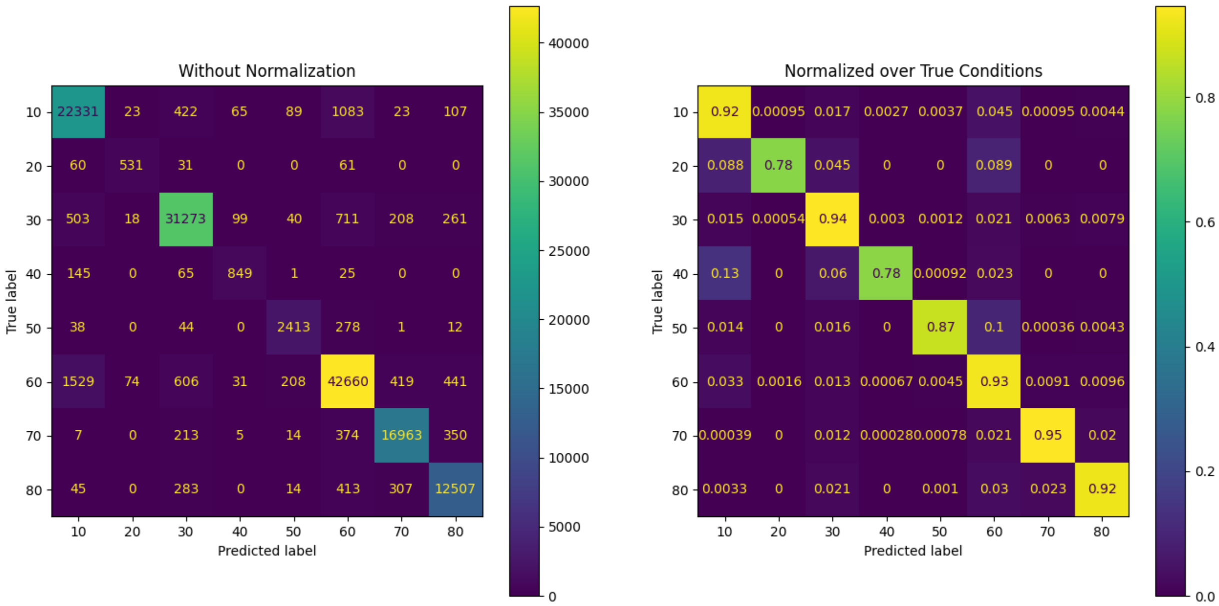

Figure 86: Confusion Matrix of IT_silver of `xlm-roberta-base_w_context_2_IT_sents`

	precision	recall	f1-score	support
macro avg	0.8939	0.8864	0.8900	139273
weighted avg	0.9301	0.9300	0.9300	139273

Test set: gold

Language: IT

- F-score (micro) 0.9014
- F-score (macro) 0.8489
- Accuracy 0.9014

By class:

	precision	recall	f1-score	support
30	0.9235	0.9095	0.9164	2442
60	0.9005	0.8758	0.8880	2077
10	0.8946	0.9411	0.9173	1461
70	0.9511	0.9258	0.9383	903
80	0.8170	0.8867	0.8504	715
50	0.8539	0.8092	0.8310	325
20	0.7391	1.0000	0.8500	17
40	0.8182	0.4737	0.6000	19

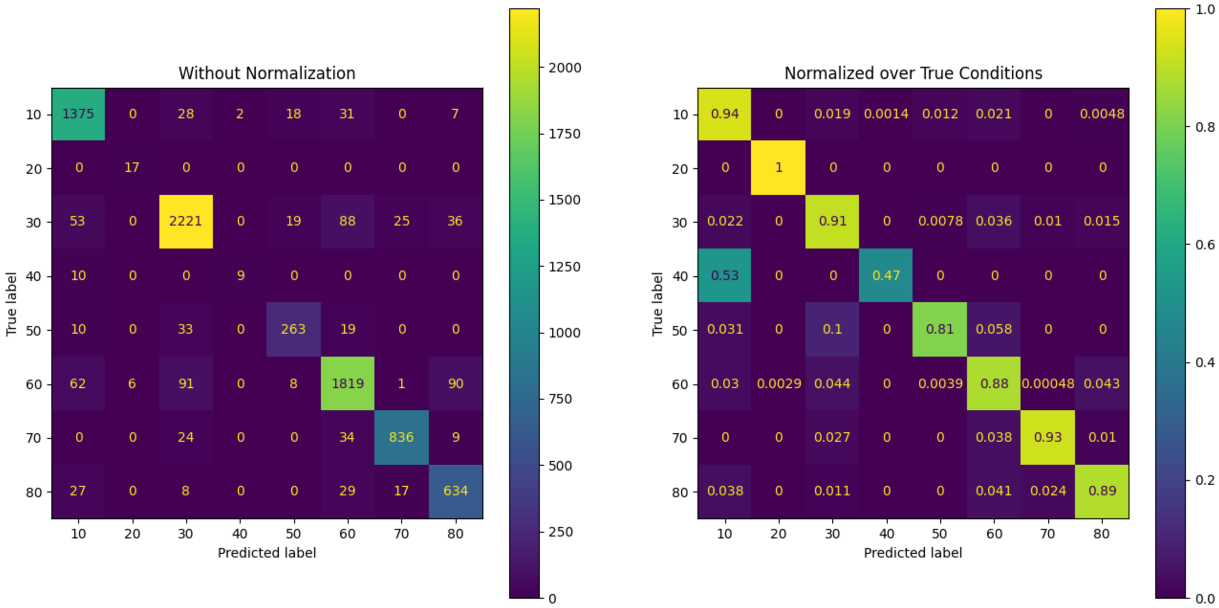

Figure 87: Confusion Matrix of IT_gold of `xlm-roberta-base_w_context_2_IT_sents`

accuracy			0.9014	7959
macro avg	0.8622	0.8527	0.8489	7959
weighted avg	0.9023	0.9014	0.9013	7959

C.14 `xlm-roberta-base_o_context_job`

Test set: silver
Language: EN-US
- F-score (micro) 0.9251
- F-score (macro) 0.8908
- Accuracy 0.9251

By class:

	precision	recall	f1-score	support
60	0.9298	0.9322	0.9310	44085
30	0.9438	0.9284	0.9360	33472
10	0.9097	0.9266	0.9180	23052
70	0.9394	0.9384	0.9389	17932

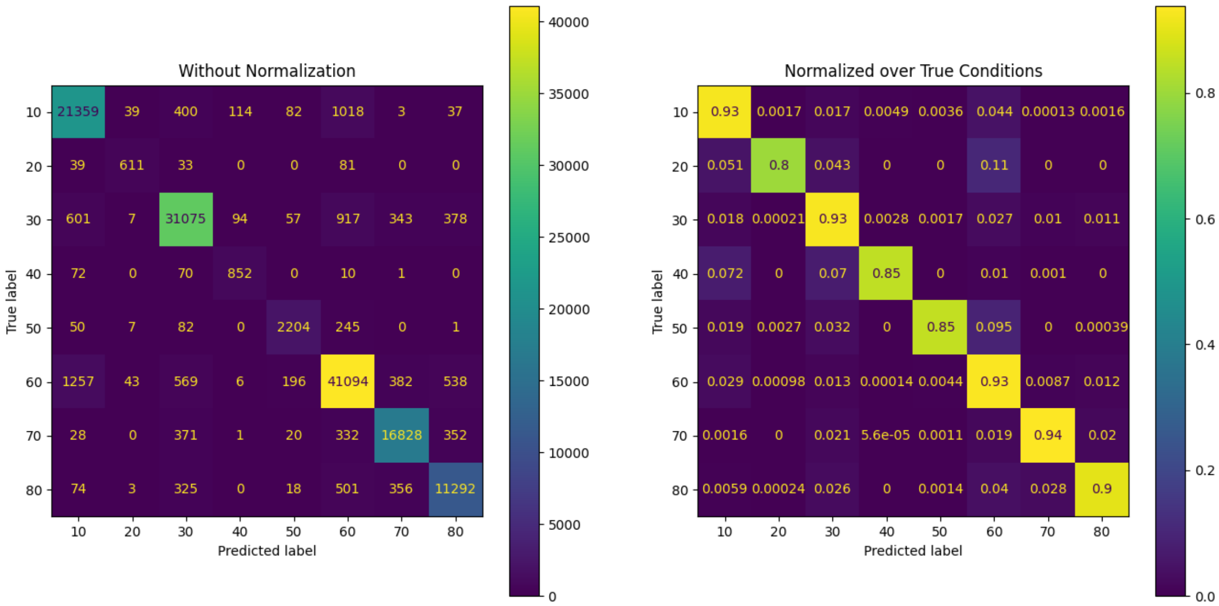

Figure 88: Confusion Matrix of EN-US_silver of `xlm-roberta-base_o_context_job`

	precision	recall	f1-score	support
80	0.8963	0.8984	0.8974	12569
50	0.8553	0.8513	0.8533	2589
40	0.7985	0.8478	0.8224	1005
20	0.8606	0.7997	0.8290	764
accuracy			0.9251	135468
macro avg	0.8917	0.8903	0.8908	135468
weighted avg	0.9252	0.9251	0.9251	135468

```
Test set: silver
Language: FR
- F-score (micro) 0.9292
- F-score (macro) 0.8945
- Accuracy 0.9292

By class:
```

	precision	recall	f1-score	support
60	0.9359	0.9322	0.9341	51052
30	0.9465	0.9333	0.9398	34599
10	0.9155	0.9306	0.9230	25856

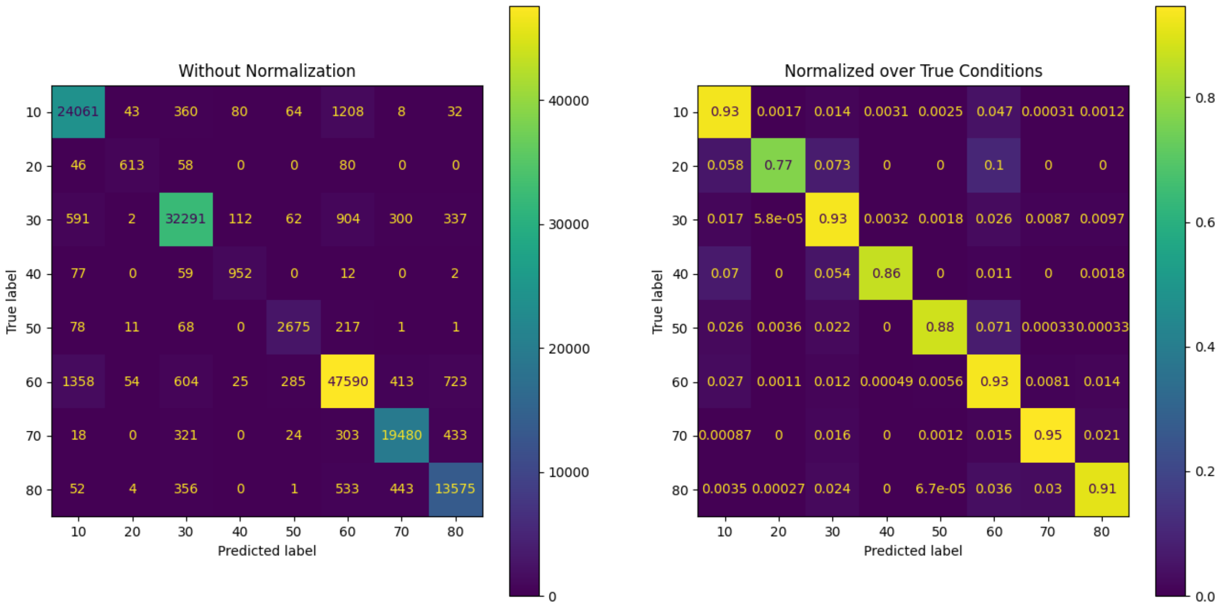

Figure 89: Confusion Matrix of FR_silver of `xlm-roberta-base_o_context_job`

	precision	recall	f1-score	support
70	0.9436	0.9466	0.9451	20579
80	0.8988	0.9072	0.9030	14964
50	0.8599	0.8768	0.8682	3051
40	0.8144	0.8639	0.8384	1102
20	0.8432	0.7691	0.8045	797
accuracy			0.9292	152000
macro avg	0.8947	0.8950	0.8945	152000
weighted avg	0.9294	0.9292	0.9292	152000

Test set: silver
Language: IT
- F-score (micro) 0.9291
- F-score (macro) 0.8968
- Accuracy 0.9291

By class:

	precision	recall	f1-score	support
60	0.9299	0.9330	0.9314	45968
30	0.9505	0.9362	0.9433	33113

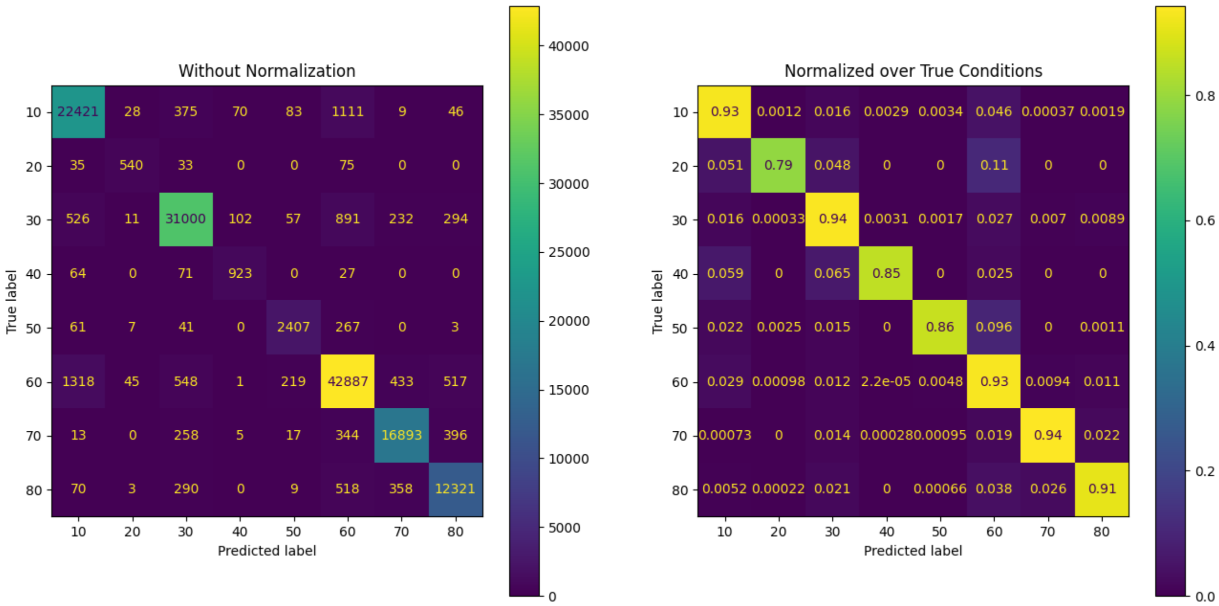

Figure 90: Confusion Matrix of IT_silver of `xlm-roberta-base_o_context_job`

```
         10      0.9148    0.9287    0.9217     24143
         70      0.9424    0.9424    0.9424     17926
         80      0.9075    0.9080    0.9078     13569
         50      0.8621    0.8640    0.8630      2786
         40      0.8383    0.8507    0.8445      1085
         20      0.8517    0.7906    0.8200       683

   accuracy                          0.9291    139273
  macro avg      0.8997    0.8942    0.8968    139273
weighted avg     0.9292    0.9291    0.9291    139273


Test set: gold
Language: EN-US
- F-score (micro) 0.9294
- F-score (macro) 0.7994
- Accuracy 0.9294

By class:
              precision    recall  f1-score   support

         60      0.9357    0.9753    0.9551      3597
```

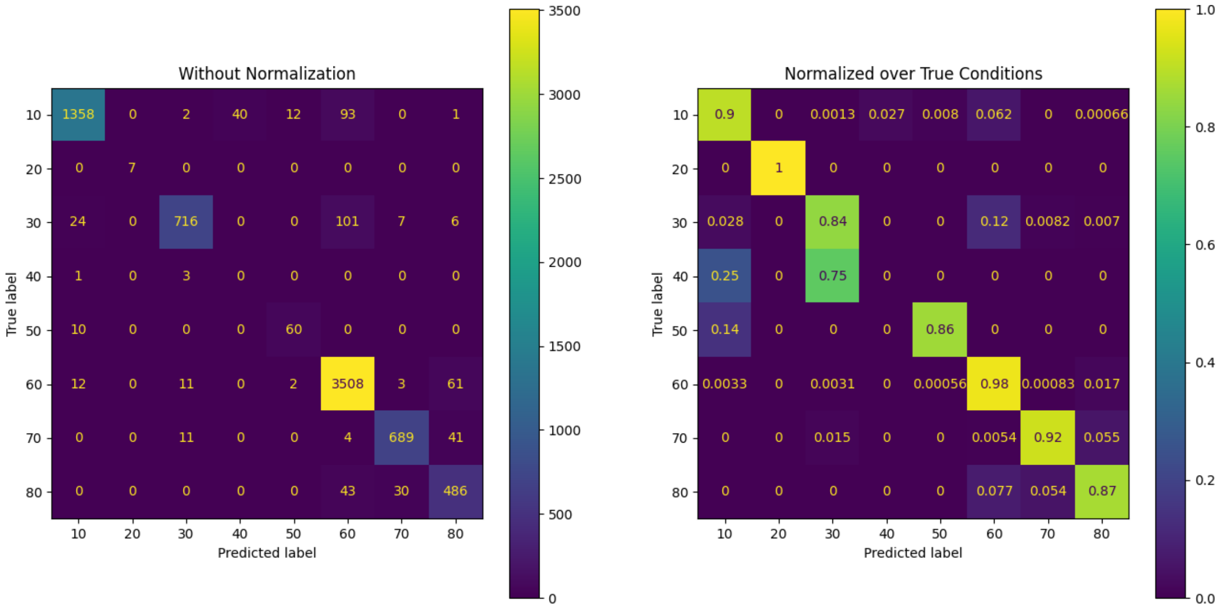

Figure 91: Confusion Matrix of EN-US_gold of `xlm-roberta-base_o_context_job`

10	0.9665	0.9017	0.9330	1506
30	0.9637	0.8384	0.8967	854
70	0.9451	0.9248	0.9349	745
80	0.8168	0.8694	0.8423	559
50	0.8108	0.8571	0.8333	70
40	0.0000	0.0000	0.0000	4
20	1.0000	1.0000	1.0000	7

accuracy			0.9294	7342
macro avg	0.8048	0.7958	0.7994	7342
weighted avg	0.9356	0.9294	0.9315	7342

Test set: gold

Language: FR

- F-score (micro) 0.9331
- F-score (macro) 0.907
- Accuracy 0.9331

By class:

	precision	recall	f1-score	support

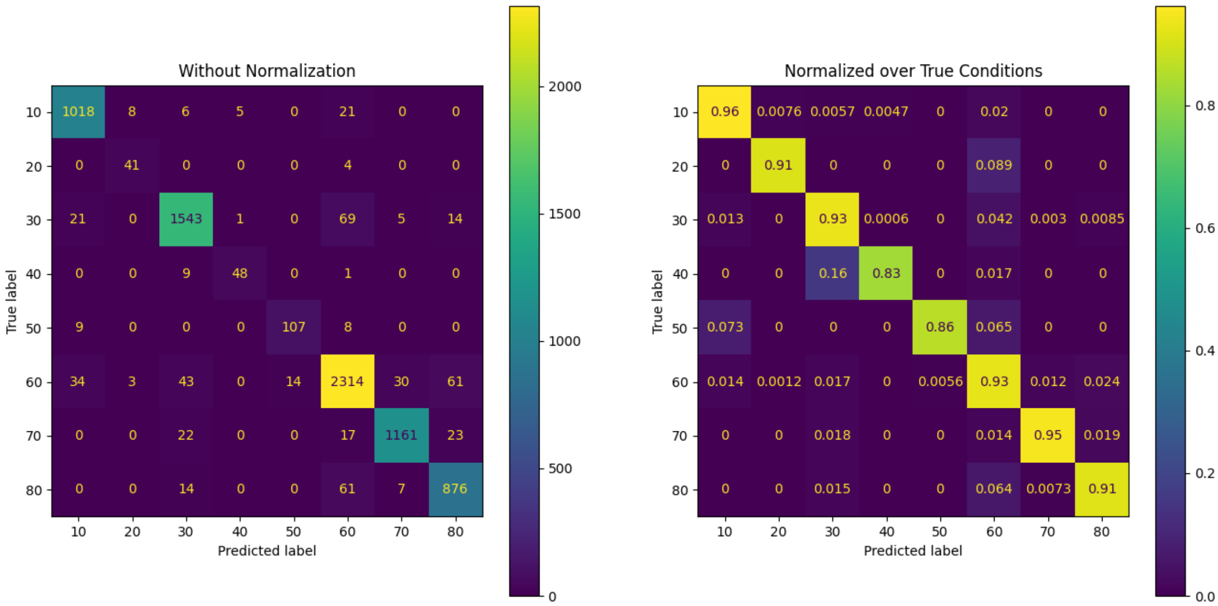

Figure 92: Confusion Matrix of FR_gold of `xlm-roberta-base_o_context_job`

60	0.9275	0.9260	0.9267	2499
30	0.9426	0.9335	0.9380	1653
70	0.9651	0.9493	0.9571	1223
10	0.9409	0.9622	0.9514	1058
80	0.8994	0.9144	0.9068	958
50	0.8843	0.8629	0.8735	124
40	0.8889	0.8276	0.8571	58
20	0.7885	0.9111	0.8454	45
accuracy			0.9331	7618
macro avg	0.9046	0.9109	0.9070	7618
weighted avg	0.9333	0.9331	0.9331	7618

```
Test set: gold
Language: IT
- F-score (micro) 0.9134
- F-score (macro) 0.8995
- Accuracy 0.9134

By class:
              precision    recall  f1-score   support
```

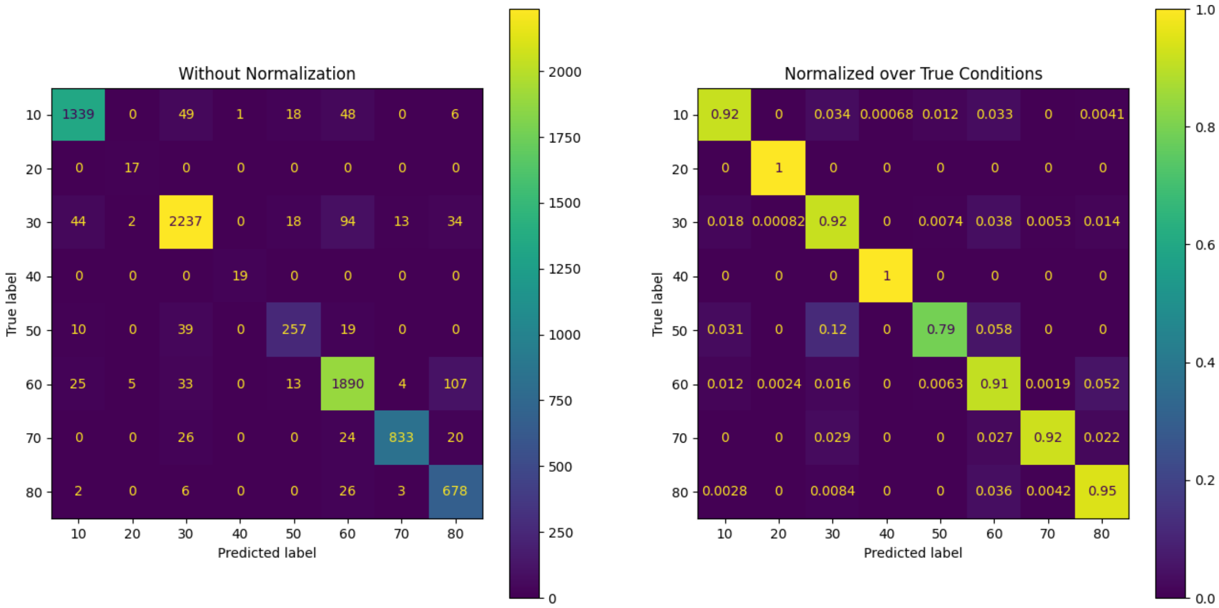

Figure 93: Confusion Matrix of IT_gold of `xlm-roberta-base_o_context_job`

30	0.9360	0.9161	0.9259	2442
60	0.8996	0.9100	0.9047	2077
10	0.9430	0.9165	0.9295	1461
70	0.9766	0.9225	0.9487	903
80	0.8024	0.9483	0.8692	715
50	0.8399	0.7908	0.8146	325
20	0.7083	1.0000	0.8293	17
40	0.9500	1.0000	0.9744	19
accuracy			0.9134	7959
macro avg	0.8820	0.9255	0.8995	7959
weighted avg	0.9160	0.9134	0.9139	7959

```
Test set: gold
Language: DE
- F-score (micro) 0.9177
- F-score (macro) 0.8802
- Accuracy 0.9177

By class:
```

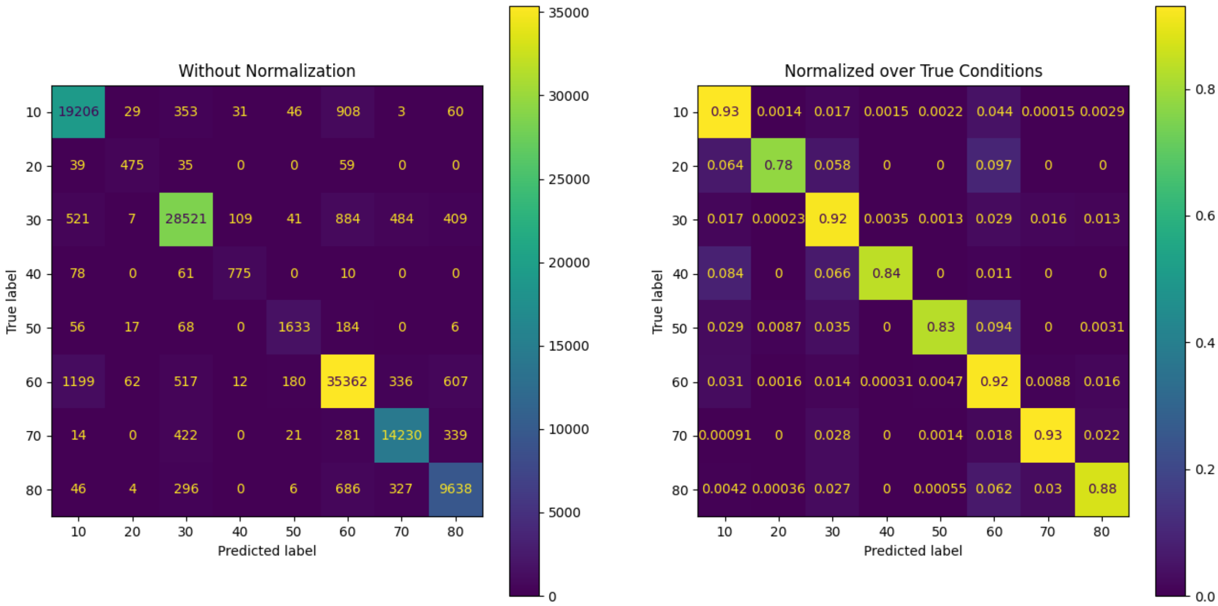

Figure 94: Confusion Matrix of DE_gold of `xlm-roberta-base_o_context_job`

	precision	recall	f1-score	support
60	0.9215	0.9239	0.9227	38275
30	0.9421	0.9207	0.9313	30976
10	0.9077	0.9307	0.9191	20636
70	0.9252	0.9296	0.9274	15307
80	0.8715	0.8759	0.8737	11003
50	0.8474	0.8315	0.8394	1964
40	0.8360	0.8387	0.8374	924
20	0.7997	0.7812	0.7903	608
accuracy			0.9177	119693
macro avg	0.8814	0.8790	0.8802	119693
weighted avg	0.9178	0.9177	0.9177	119693

C.15 `xlm-roberta-base-job`

Test set: silver

Language: EN-US

- F-score (micro) 0.9243

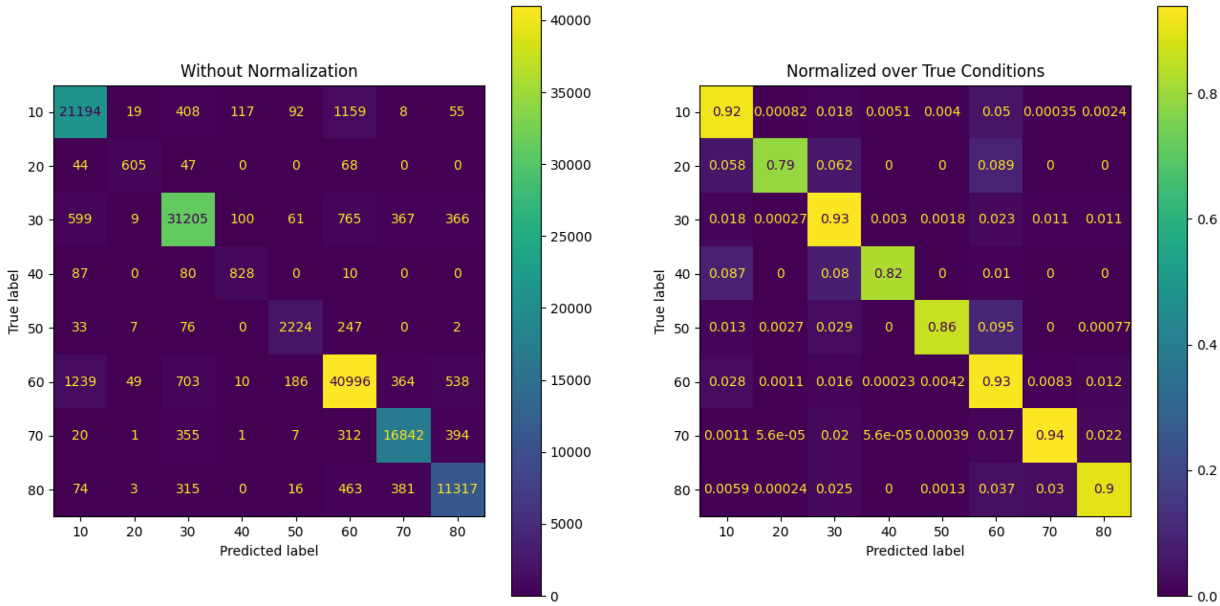

Figure 95: Confusion Matrix of EN-US_silver of `xlm-roberta-base-job`

```
- F-score (macro) 0.8888
- Accuracy 0.9243

By class:
              precision    recall  f1-score   support

          60     0.9313    0.9299    0.9306     44085
          30     0.9402    0.9323    0.9362     33472
          10     0.9100    0.9194    0.9147     23052
          70     0.9376    0.9392    0.9384     17932
          80     0.8931    0.9004    0.8967     12569
          50     0.8600    0.8590    0.8595      2589
          40     0.7841    0.8239    0.8035      1005
          20     0.8730    0.7919    0.8305       764

    accuracy                         0.9243    135468
   macro avg     0.8912    0.8870    0.8888    135468
weighted avg     0.9244    0.9243    0.9243    135468

Test set: silver
Language: FR
```

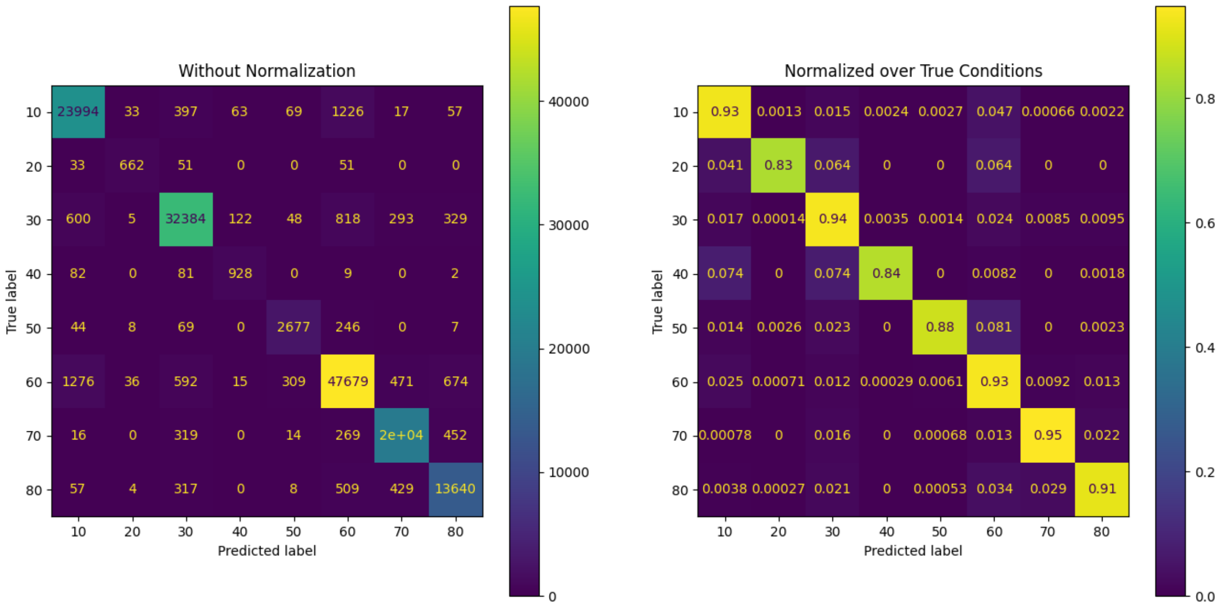

Figure 96: Confusion Matrix of FR_silver of `xlm-roberta-base-job`

```
- F-score (micro) 0.9307
- F-score (macro) 0.9009
- Accuracy 0.9307

By class:
              precision    recall  f1-score   support

          60     0.9384    0.9339    0.9362     51052
          30     0.9466    0.9360    0.9413     34599
          10     0.9192    0.9280    0.9236     25856
          70     0.9416    0.9480    0.9448     20579
          80     0.8997    0.9115    0.9056     14964
          50     0.8566    0.8774    0.8669      3051
          40     0.8227    0.8421    0.8323      1102
          20     0.8850    0.8306    0.8570       797

    accuracy                         0.9307    152000
   macro avg     0.9012    0.9009    0.9009    152000
weighted avg     0.9309    0.9307    0.9308    152000


Test set: silver
```

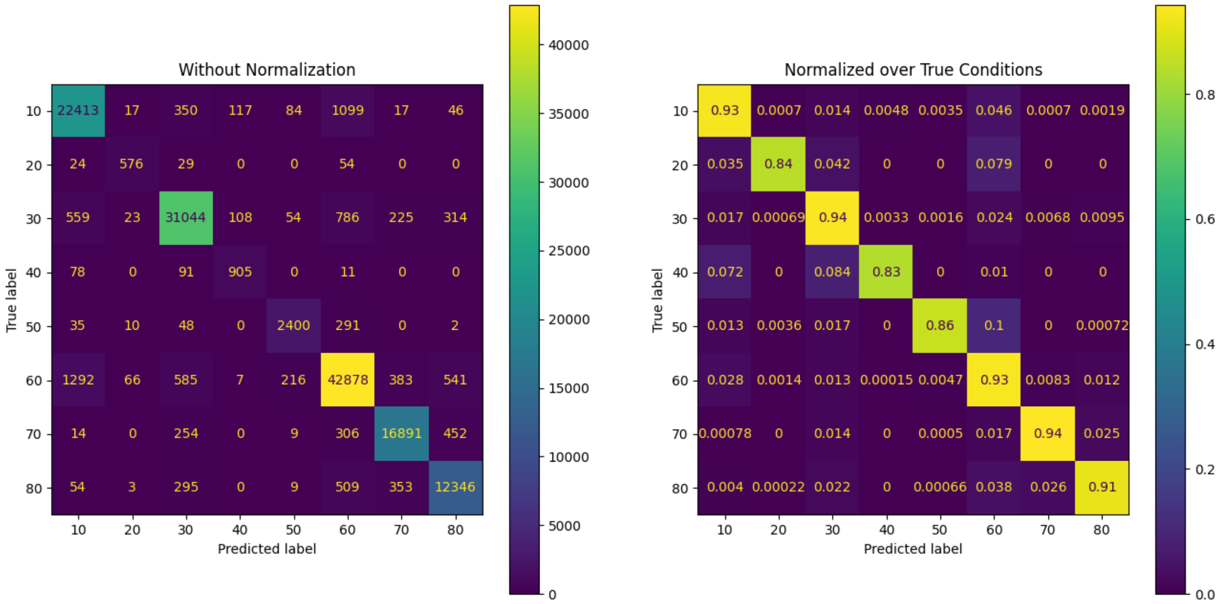

Figure 97: Confusion Matrix of IT_silver of `xlm-roberta-base-job`

```
Language: IT
- F-score (micro) 0.9295
- F-score (macro) 0.8953
- Accuracy 0.9295

By class:
              precision    recall  f1-score   support

          60     0.9335    0.9328    0.9331     45968
          30     0.9495    0.9375    0.9435     33113
          10     0.9160    0.9283    0.9221     24143
          70     0.9453    0.9423    0.9438     17926
          80     0.9011    0.9099    0.9055     13569
          50     0.8658    0.8615    0.8636      2786
          40     0.7960    0.8341    0.8146      1085
          20     0.8288    0.8433    0.8360       683

    accuracy                         0.9295    139273
   macro avg     0.8920    0.8987    0.8953    139273
weighted avg     0.9297    0.9295    0.9296    139273
```

```
Test set: gold
Language: EN-US
- F-score (micro) 0.9323
- F-score (macro) 0.7724
- Accuracy 0.9323

By class:
              precision    recall  f1-score   support

          60     0.9447    0.9689    0.9566      3597
          10     0.9526    0.8805    0.9151      1506
          30     0.9497    0.8630    0.9043       854
          70     0.9493    0.9544    0.9518       745
          80     0.8614    0.9338    0.8961       559
          50     0.8000    0.8571    0.8276        70
          40     0.0000    0.0000    0.0000         4
          20     1.0000    0.5714    0.7273         7

    accuracy                         0.9323      7342
   macro avg     0.8072    0.7536    0.7724      7342
weighted avg     0.9392    0.9323    0.9350      7342


Test set: gold
Language: FR
- F-score (micro) 0.9295
- F-score (macro) 0.9014
- Accuracy 0.9295

By class:
              precision    recall  f1-score   support

          60     0.9427    0.9280    0.9353      2499
          30     0.9415    0.9341    0.9377      1653
          70     0.9497    0.9109    0.9299      1223
          10     0.9433    0.9442    0.9438      1058
          80     0.8657    0.9489    0.9054       958
          50     0.8088    0.8871    0.8462       124
          40     0.7931    0.7931    0.7931        58
```

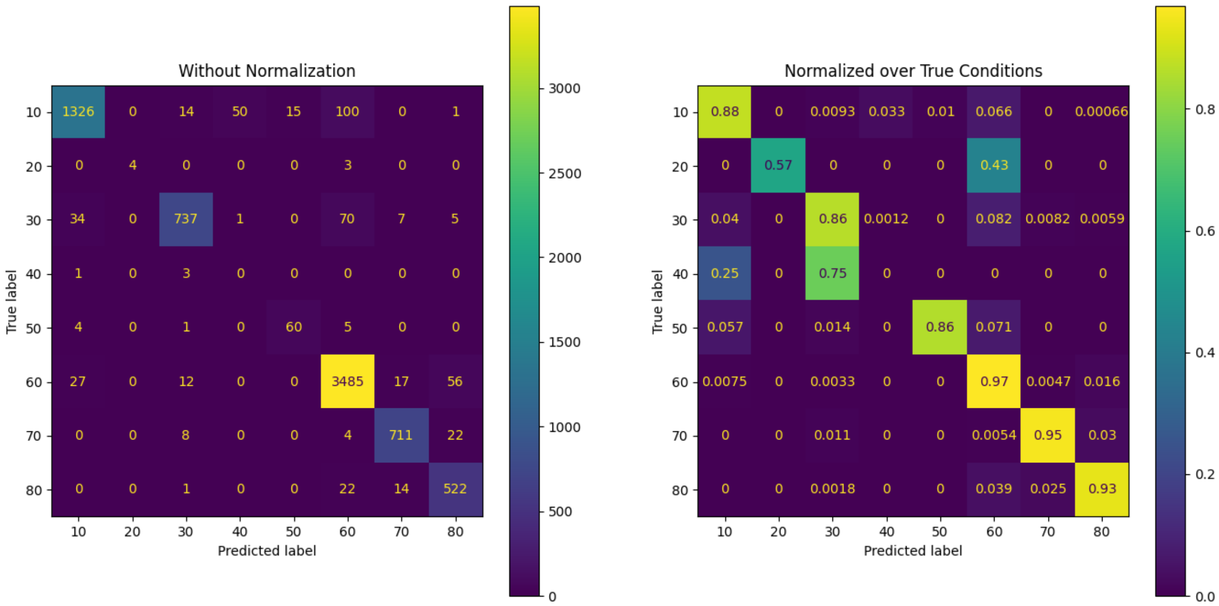

Figure 98: Confusion Matrix of EN-US_gold of `xlm-roberta-base-job`

	precision	recall	f1-score	support
20	0.9524	0.8889	0.9195	45
accuracy			0.9295	7618
macro avg	0.8997	0.9044	0.9014	7618
weighted avg	0.9307	0.9295	0.9297	7618

```
Test set: gold
Language: IT
- F-score (micro) 0.9211
- F-score (macro) 0.9129
- Accuracy 0.9211

By class:
```

	precision	recall	f1-score	support
30	0.9479	0.9312	0.9395	2442
60	0.9084	0.9023	0.9053	2077
10	0.9160	0.9254	0.9207	1461
70	0.9803	0.9358	0.9575	903
80	0.8273	0.9315	0.8763	715
50	0.9161	0.8738	0.8945	325

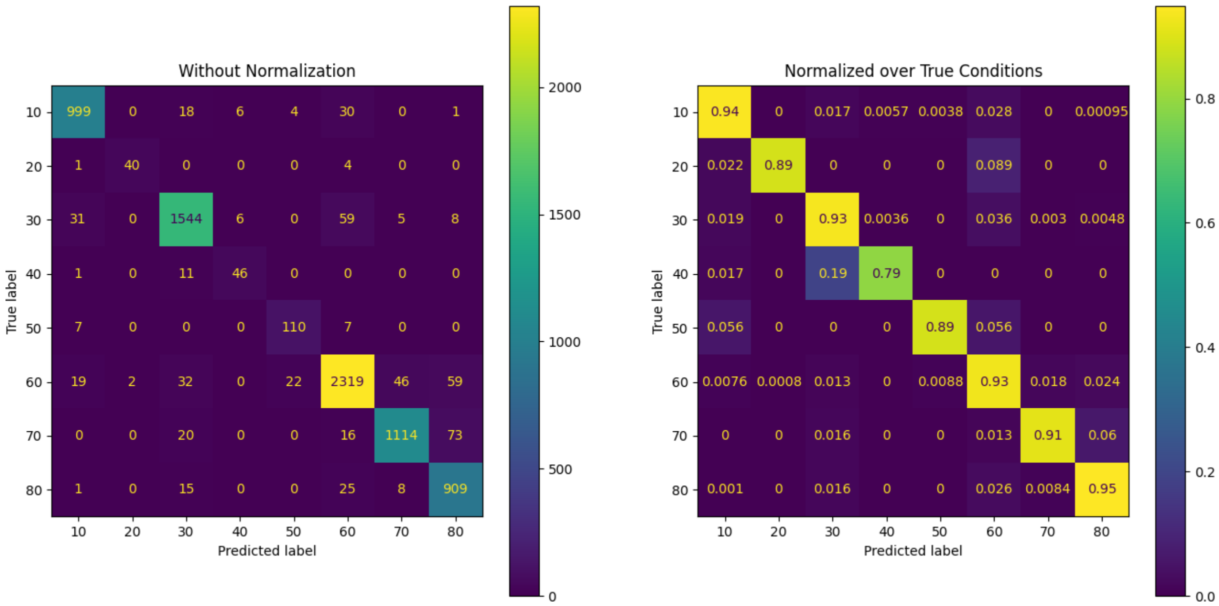

Figure 99: Confusion Matrix of FR_gold of `xlm-roberta-base-job`

20	0.6800	1.0000	0.8095	17
40	1.0000	1.0000	1.0000	19
accuracy			0.9211	7959
macro avg	0.8970	0.9375	0.9129	7959
weighted avg	0.9228	0.9211	0.9215	7959

```
Test set: gold
Language: DE
- F-score (micro) 0.9202
- F-score (macro) 0.8847
- Accuracy 0.9202
```

By class:

	precision	recall	f1-score	support
60	0.9265	0.9256	0.9261	38275
30	0.9422	0.9218	0.9319	30976
10	0.9139	0.9318	0.9228	20636
70	0.9227	0.9347	0.9287	15307
80	0.8716	0.8819	0.8767	11003

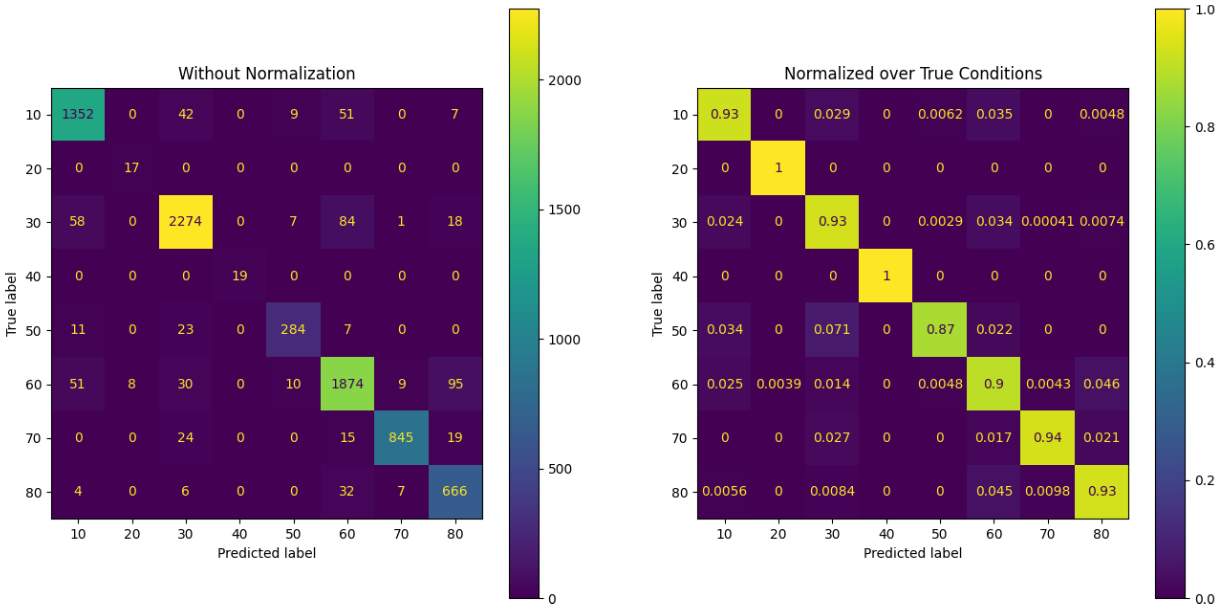

Figure 100: Confusion Matrix of IT_gold of `xlm-roberta-base-job`

50	0.8515	0.8498	0.8507	1964
40	0.8270	0.8431	0.8349	924
20	0.8345	0.7796	0.8061	608
accuracy			0.9202	119693
macro avg	0.8862	0.8835	0.8847	119693
weighted avg	0.9204	0.9202	0.9202	119693

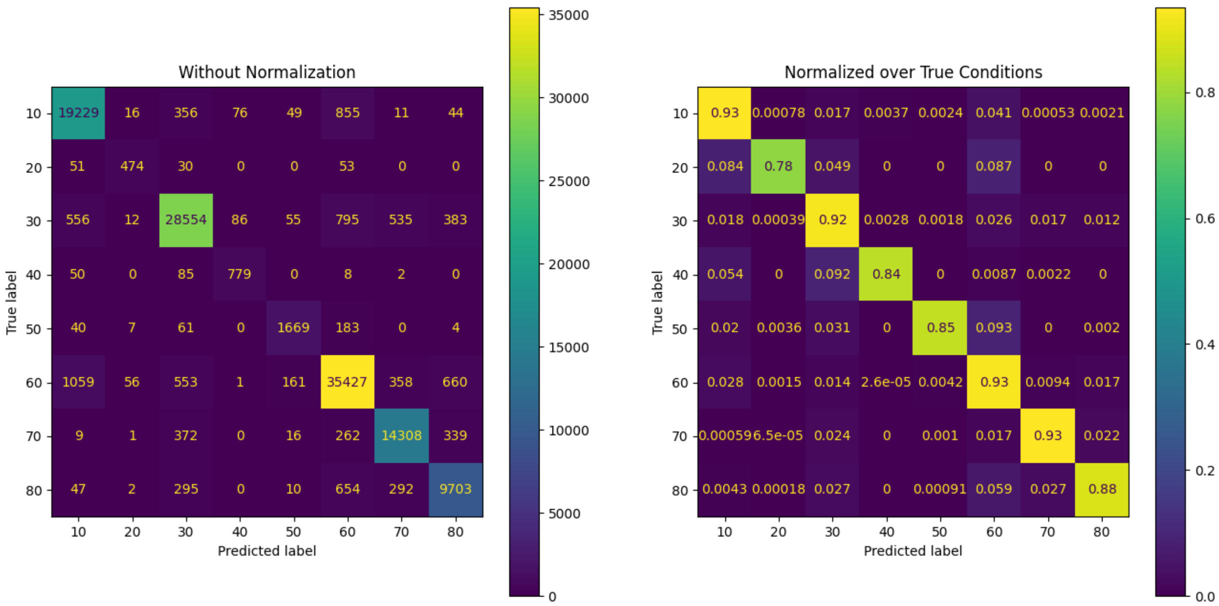

Figure 101: Confusion Matrix of DE_gold of `xlm-roberta-base-job`