Universidade de Lisboa

Faculdade de Letras

# NAMED ENTITY ERROR PREDICTION ACROSS DOMAINS

Mestrado em Tradução

JULIANA VALPASSO DE ANDRADE

2024

Relatório de estágio especialmente elaborado para a obtenção do grau de Mestre em Tradução, orientado pela professora Doutora Helena Gorete Silva Moniz e coorientado pela Doutora Vera Mónica dos Santos Cabarrão

Para os professores Carlos Eduardo Valencia Villa

e Ana Teresinha Miranda dos Guaranys,

por me apresentarem novos horizontes

e possibilidades.

## ACKNOWLEDGMENTS

Às minhas orientadoras, a professora Doutora Helena Gorete Silva Moniz e a Doutora Vera Mónica dos Santos Cabarrão, agradeço por todo o conhecimento compartilhado ao longo dessa jornada. Esse trabalho não existiria sem o suporte de vocês.

À Unbabel, por me receber de portas gentilmente abertas e por me proporcionar uma experiência excepcional de estágio, que tanto contribuiu para o meu crescimento profissional e pessoal.

Aos meus pais, Rosimere e David, por sempre acreditarem e apoiarem os meus sonhos mais loucos. Se hoje tenho asas para voar, eu devo isso à vocês.

À Rodolpho, meu companheiro de aventuras e desventuras da vida. Obrigada por estar sempre presente e por ter tanta coragem para compartilhar comigo essa jornada.

Aos queridos Lícia e Fernando, minha família em Portugal. Obrigada por todas as conversas, conselhos e atenção que vocês têm, tão gentilmente, me dado.

À Jingxuan Yan, que se tornou uma amiga e uma grande companheira para momentos de risos e de desespero. A experiência de estágio não teria sido a mesma sem você.

*Quem, de três milênios,*
*Não é capaz de se dar conta*
*Vive na ignorância, na sombra,*
*À mercê dos dias, do tempo.*

Johann Wolfgang von Goethe

# TABLE OF CONTENTS

# ABSTRACT

This thesis was conducted within the scope of Machine Translation (MT) as part of an internship at the Portuguese company Unbabel, a startup that uses MT and other Artificial Intelligence (AI) services. In this work we propose to analyse the Named Entity Error Prediction (NEEP) model developed by the company, which also involves a community of translators contributing to error annotation and post-editing of MT.

Over the past decade, much research has focused on Named Entities (NE), whether in the domain of Named Entity Recognition (NER) systems for their identification (Nouvel, 2016) or within the Multidimensional Quality Metrics (MQM) framework (Lommel, 2014), with the goal of classifying them accurately to enhance MT processes for NEs. Therefore, the main objective of this thesis is to analyse the first version of the NEEP model (v1) and, based on this analysis, explore paths for improving the model through data augmentation to retrain it. This process led to the development of a new model version, known as NEEP v2, which underwent testing and demonstrated substantial performance improvements.

Understanding the process of NE annotation and error annotation in NEs was a fundamental part of the research methodology, which is divided into two parts. The first part entails describing the two datasets used first for training the initial version of the NEEP model and, subsequently, for testing it. The second part details the data augmentation process, focusing on constructing a new dataset for retraining the NEEP model to address the most common NE errors.

From the results we obtained throughout the first analysis, we gained insights into the main areas requiring improvement in the NEEP (v1) model. Building a dataset for data augmentation primarily involved inducing errors in the NEs that were most error-prone. In this regard, we used data from different clients in various domains and languages to generate these new errors, using *smaug* - a multilingual data augmentation package that focuses on altering specific aspects of sentences, such as NEs - and manual induction of NE errors. After the retraining phase, we obtained a new version, called NEEP v2, and subsequently selected a sample of 3,164 segments to test the new model.

Considering it was our initial attempt to create a model capable of identifying NE errors in MT outputs, we believe we achieved successful outcomes. We successfully addressed

significant NE error issues across various languages and domains, as evidenced by our initial assessment of the NEEP v1 model and the subsequent analysis of the NEEP v2 model following the data augmentation we proposed.

In conclusion, we believe that the study of both versions of the NEEP model has significantly contributed to Unbabel. All the research process led to the development of a new model version (NEEP v2), that is able to work in quite different language pairs and domains. Its prominent F-measure results (over 20 F-measure increased performance) also represent a remarkable gain and sets light for further investigations towards achieving increasingly satisfactory model results.

**Key words:** Named Entity; Named Entity Error Prediction; Named Entity Recognition (NER); Error Annotation; Multidimensional Quality Metrics Framework.

# RESUMO

Esta dissertação foi realizada no âmbito da Tradução Automática (TA), como parte das atividades de estágio na empresa portuguesa Unbabel, uma *startup* que utiliza TA e outros serviços de Inteligência Artificial (IA). O principal objetivo do presente trabalho é analisar o módulo para predição de erros em entidades mencionadas (Named Entity Error Prediction), desenvolvido pela empresa, que também conta com uma comunidade de tradutores que contribuem para o processo de anotação de erros e pós-edição das TAs.

Desde a última década, muitos estudos têm-se centrado em análises de Entidades Mencionadas (EM) - seja na área de sistemas de Reconhecimento de Entidades Mencionadas (REM) para as identificar (Nouvel, 2016), seja dentro do contexto das *Multidimensional Quality Metrics* (MQM) (Lommel, 2014), com o objetivo de classificá-las adequadamente quanto a erros de tradução, visando melhorar os processos de TA para as EMs.

No que diz respeito ao reconhecimento de EMs, um sistema REM eficaz deve ser capaz de identificar diversas EMs num texto de partida e, em seguida, classificá-las de acordo com categorias pré-definidas. Os sistemas REM desempenham um papel fundamental como ponto de partida no processo de compreensão do significado de um texto e são essenciais para diversas outras tarefas de Processamento de Linguagem Natural (PLN), incluindo Anonimização, Localização e previsão de erros em EMs, que são o foco desta pesquisa.

Por conseguinte, o objetivo principal desta dissertação consiste em analisar a primeira versão do modelo NEEP (v1) e, a partir dessa análise, investigar as possibilidades de melhoria do sistema por meio do aumento de dados para retreinar o modelo. Esse processo culminou na criação de uma nova versão do modelo, denominada NEEP v2, que foi sujeita a testes e apresentou melhorias consideráveis no seu desempenho.

Não obstante, é preciso reiterar que para chegar ao processo de análise dos pontos que precisavam de melhorias no modelo e assim, prover novos dados para que o modelo fosse retreinado e avaliado novamente, foi elaborado um breve apanhado histórico a respeito do desenvolvimento da TA, bem como das investigações feitas sobre EMs até ao presente momento. Consideramos importante estabelecer como os sistemas de REM funcionam e as principais razões para termos um sistema *in-house*, no que diz respeito às EMs de clientes e domínios específicos com os quais lidamos no dia a dia da Unbabel. Ter um sistema próprio que trabalha

com categorias específicas de EMs requer uma atenção especial quanto às próprias entidades e, consequentemente, os processos de reconhecimento, anotação, anonimização e localização.

Entender como funciona o processo de anotação das EMs, bem como o processo de anotação de erros nas EMs, fez parte fundamental da metodologia do trabalho, que está dividida em duas partes. A primeira corresponde à descrição dos bancos de dados que foram usados primeiro para treinar a primeira versão do modelo NEEP e, segundo, para testá-lo. A segunda parte descreve o processo de ampliação de dados, com foco na construção de um novo banco de dados para retreinar o modelo NEEP, a fim de abordar os erros mais comuns nas EMs.

Para que esse novo banco de dados para ampliação dos dados fosse construído, realizamos uma análise detalhada dos resultados obtidos a partir do primeiro teste do modelo. Essa análise contou com o processo de anotação de erros em EMs e, a partir dessa anotações, foi possível estabelecer não apenas os valores de *precision* e *recall* - 0.9965 e 0.3666, respectivamente - como também as categorias de erros mais recorrentes e em quais línguas o modelo falhava mais na identificação dos erros em EMs.

Dentre as categorias de erros em EMs identificadas como mais prováveis de o modelo falhar, encontramos *Wrong NE*, com 51.28% dos segmentos contendo erros em EMs que não foram identificados pelo modelo NEEP, e *Date/time format*, com 3.89% de erros em EMs. No que diz respeito às línguas mais propensas a erros em EMs que o modelo mais falha, encontramos árabe e coreano, línguas não provenientes de matriz latina, e francês, dentro da matriz latina.

A partir desses resultados, pudemos compreender os principais pontos que precisavam de melhorias no modelo NEEP. A construção de um banco de dados para ampliação dos dados para retreinar o modelo partiu principalmente da indução de erros nas EMs que eram mais propensas a erros. Nesse sentido, utilizamos dados de diferentes clientes em diferentes domínios para gerar esses novos erros, utilizando *smaug*[1] - um pacote de ampliação de dados multilíngue que oferece alterações focadas em modificar aspectos específicos de frases, como as EMs - e indução manual de erros em EMs. Em um sentido mais pragmático, o script automaticamente substitui as EMs do texto de partida por outras EMs, gerando um conjunto de perturbações que utilizamos para retreinar o modelo NEEP. Após a fase de retreino do modelo, obtivemos uma nova versão,

[1] Available at: <https://pypi.org/project/unbabel-smaug/#description>.

chamada NEEP v2. Em seguida, selecionamos uma amostra de 3,164 segmentos para testar o novo modelo.

Em comparação com o modelo NEEP v1, o novo modelo - NEEP v2 - teve um aumento considerável no valor de *recall*, passando de 0.3666 no v1 para 0.6982 no v2. O mesmo aconteceu com o valor de *F-measure*, que passou de 0.5360 para 0.7701, o que representa um aumento de mais de 20 pontos. No que diz respeito à *precision*, embora o valor tenha sofrido uma leve queda, de 0.9965 para 0.8584, o modelo NEEP v2 ainda manteve um nível elevado de precisão na identificação dos erros em EMs.

Dentre os aspectos mais marcantes da melhoria da segunda versão do modelo, destacamos as categorias de EMs *Wrong NE*, *Date/time format* e *Address format*. No total de erros em EMs, o modelo identificou 80.89% para *Wrong NE*, 8.88% para *Date/time format* e 0.71% para *Address format* - cuja primeira versão do modelo não havia identificado quaisquer dos erros nessa categoria. O modelo NEEP v2 também apresentou melhorias consideráveis nas línguas com as quais foi testado, especialmente árabe e coreano que não são de matriz latina, e húngaro e francês de matriz latina. Ainda que tenha havido determinadas falhas em categorias como *Measurement format*, por exemplo, o desempenho global do modelo é excepcional.

Por fim, consideramos que o estudo com as duas versões do modelo NEEP contribuíram de maneira significativa para a empresa. O desenvolvimento de uma nova versão do modelo que apresenta resultados de *F-measure* proeminentes representa um ganho considerável e abre portas para a possibilidade de que novas investigações continuem sendo feitas em prol de obter um modelo com resultados cada vez mais satisfatórios.

**Palavras-chave:** Entidades Mencionadas; Predição de Erros em Entidades Mencionadas (NEEP); Reconhecimento de Entidades Mencionada (REM); Anotação de erros; *Multidimensional Quality Metrics Framework*.

# LIST OF TABLES

# LIST OF FIGURES

# 1. INTRODUCTION

The present thesis focuses on Named Entity (NE) errors in Neural Machine Translation (NMT) and was completed during an internship at Unbabel, a Portuguese software company specialising in providing translation services for areas such as customer support, marketing, press releases and travel. The main objective of this thesis is to assess the NEEP model and enhance its performance through data augmentation. This entails expanding the existing dataset of NE errors by incorporating additional errors from different clients, domains, and content types. Thus we intend to evaluate whether the data augmentation led to improved error prediction across various languages and domains, preventing critical NE errors.

Much research has been carried out about NE, whether in the Named Entity Recognition (NER) systems scope, concerning their identification, or in the MQM framework, with the effort of properly classifying errors related with them, in order to enhance the machine translation (MT) procedures towards NE (Nouvel, 2016). With respect to NE recognition, a proficient NER system should possess the ability to recognize distinct NEs in a given source text and, subsequently, categorise them based on predefined categories (Oliveira, 2010). NER systems are important as they serve as an initial step in the process of understanding the meaning of a text. They are crucial facilitators for various other Natural Language Processing (NLP) tasks, including Anonymization, Localization and NE error prediction, which is the aim of this research.

Accordingly, though MT systems provide efficient and cost-saving translations, they might also cause errors. Some of those errors might not represent a major damage to the translation process. Nevertheless, when we turn our attention to NEs, we must take precautionary steps, since we are dealing with Personal Identifiable Information (PII). Thus, what mainly motivated the present work was better understanding and dealing with NEs, so as to prevent errors and, as much as possible, increase the quality of the translation process. We not only tackle the comprehension of Unbabel's NER system, but also analyse and improve the Named Entity Error Prediction model (NEEP).

The NEEP model is an automated system designed for predicting NE errors. The aim is to have a model able to identify possible errors after the MT step; although it may also be used during the post-editing (PE) process that involves entities. The presence of a model for NE error

prediction can reduce the reliance on human annotators, resulting in decreased time and cost expenses throughout the translation process.

In the scope of the present work, we will analyse the NEEP model developed at Unbabel in order to evaluate its strengths and weaknesses in predicting NE errors. Analysing the model's performance will provide us a more comprehensive understanding of the NE recognition and annotation processes, carried out in the source texts, as well as the most error prone NEs, that might need further attention and care.

Additionally, from the results of the first experiments, we expect to be able to create a data augmentation dataset, based on the most recurrent NE errors in the MT output, also taking into consideration the language pairs. It is expected that this analysis will enhance the retrain of the NEEP model by identifying prevalent errors and devising methods to address and rectify them within the model.

Following the thesis structure, in Chapter 2 we will mainly discuss the processes carried out at Unbabel, focusing on the NLP team and its products, such as the NER system and the NEEP model, which is the core of this work.

In Chapter 3, we intend to present the state of the art in some translation domains, such as MT history, NER system history, as well as the relation between it, the NEs and the NEEP model. We also discuss the assessment quality metrics and the performance metrics, used to evaluate the NEEP model performance before and after the data augmentation and retraining phases.

Chapter 4 will be dedicated to the methodology part. This chapter will be divided into two experiments carried out along this work. The first one is the assessment of the NEEP model, considering its primary training and testing phases. The subsequent experiment explores the data augmentation process, with a particular emphasis on constructing a new dataset aimed at retraining the NEEP model to properly address the prominent NE errors.

In Chapter 5 we will present the results of the work. As in Chapter 4, this chapter will be divided into two parts: the first will tackle the analysis of the first version of the NEEP model. It will be an important step of the work, because we will be able to identify the strengths and the weaknesses of the model and, afterwards, we expect to create a data augmentation dataset to retrain it. After the data augmentation process, we intend to test the NEEP model again and

reassess its outputs, in order to analyse how successful we were in retraining the NE errors prediction model.

Finally, in Chapter 6 we will present the final conclusions of this work along with a reflection about the possibilities of future work that might be developed based on the work carried out during this internship.

# 2. HOST INSTITUTION

## 2.1. UNBABEL'S CHARACTERIZATION

Founded in 2013, Unbabel is a translation company whose main goal is to provide machine translation and post-editing for several domains, such as customer support, marketing, press releases and travel. To make it possible, Unbabel draws upon a combination of machine translation and human post-editing in order to deliver a fast and proper translation for its clients. According to Graça (2018), the company is able to put together the speed of a machine translation with the experience of a native human translator.

Unbabel is integrated with several cloud-based Customer Relationship Management (CRM) platforms, such as Zendesk and Salesforce. Since the main focus is customer service, Unbabel provides support to clients and end users in their own native languages, through a combination of human translator and artificial intelligence (AI) by multilingual support. In that sense, the company's aim is to offer a fast and high quality translation service, answering the clients demands as efficiently as possible.

As it has been mentioned before, Unbabel's main work is in the customer support area. This support revolves around the translation not only of tickets (emails), FAQs and real time chats, but also marketing services, for instance. The service offered in the translation process is almost the same for all the content-type documents, although there are some exceptions depending on the service level chosen by the client with variations in speed and the minimum quality requirements.

In the following section the company's workflow and the pipeline for the service-type offered by the company will be described.

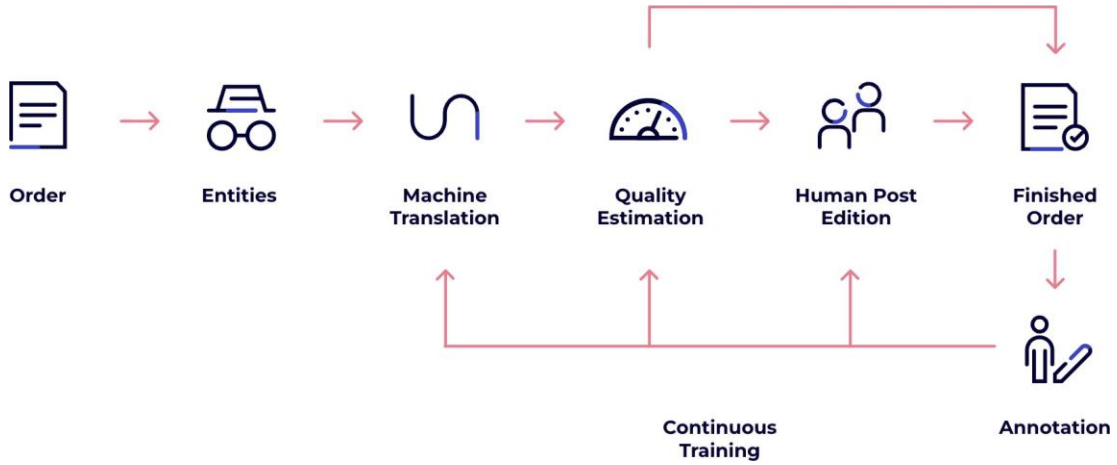
## 2.2. UNBABEL'S WORKFLOW

When analysing the company's pipeline, one of the most interesting features that can be pointed out is that, although there is a general pipeline to guide all the service, it can be adapted to suit each service level, depending on the clients' demands. This means that, on the one hand,

there are many common NLP modules for all the services offered by the company; yet, on the other hand there are steps that can, and in some cases must, be avoided, in order to provide exactly what the clients ask for, depending on the level chosen by them.

Accordingly, there are three levels of services the clients can choose from: *Realtime*, *Hybrid* and *Specialist*. The first level presented in the section is the *Hybrid* one. Although it was chosen to be presented as number one, it is not necessarily the most basic level. Nevertheless, it is important to begin with the *Hybrid* pipeline, because it contains all the steps that are required in a translation process at Unbabel.

**Figure 1 - Hybrid Translation Pipeline**




Source: Usage authorised by Unbabel, content presented by Phill Brougham.

Figure 1 illustrates the *Hybrid* translation pipeline. As it can be seen, the first step is the order placement. Once an **Order** commission is sent through one of the CRM platforms (e.g.: Zendesk, Salesforce), it is converted into a document format that can be read by Unbabel's MT system. After that, it is segmented sentence by sentence, creating what are called "nuggets". Then, all the confidential information is anonymized, in order to guarantee the privacy and the safety of the clients in a data anonymization step. Following the "General Data Protection Regulations (GDPR)[2], all Personally Identifiable Information (PII), such as proper names, emails, phone number or passwords, are identified by a Named Entity Recognition (NER) system and temporarily replaced by a placeholder" (Paulo, 2022: 12).

[2] https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679&from=EN

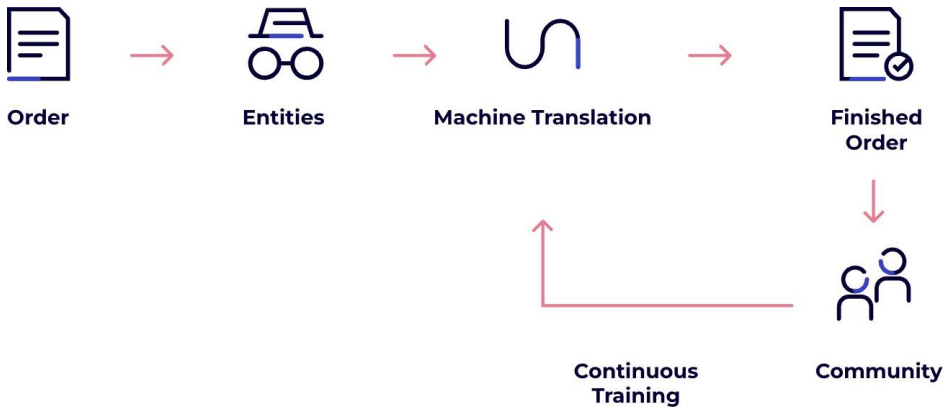
NER systems enable the identification of Named Entities (NE) by predicting the NE according to its surroundings and categorising it. After the NER systems detect a NE, it is either automatically prevented from translation or automatically marked as a NE of interest for subsequent tasks such as localization. The identification of NE by NER systems is of critical importance because it prevents several machine translation errors, such as hallucinations or mistranslations for instance, thus decreasing errors that could impact the quality of translations.

After the identification and anonymization of NE, the document goes straight to the **Machine Translation**. In this module, the customer's demands and specifications are followed and the document is translated by a machine translation engine adapted to the customer data. The subsequent module concerns the **Quality Estimation** (QE). QE has a pre-defined score threshold. If the translated document scores over that threshold, it is directly sent to the client. However, if it gets a lower score, it continues in the pipeline and goes to the next step. It is important to keep in mind that the score also depends on the client's chosen level of quality.

If it is considered that the translation needs to be edited, it is, then, sent to the **Translator Community**. The community is formed by bilingual professional and non-professional editors whose work is to analyse and correct faulty machine translations. Those editors are proficient in a certain pair of languages and their main work concerns not only guaranteeing a final product of superior quality, characterised by precision, readability, and personalised to meet specific stylistic demands, but also augmenting the Translation Memory (TM) so as to make it possible to reuse segments. The stage **Finished Order** is the last step and can also be called the **Rebuild Stage**. It means the document is restored to its initial format in the target language and all the anonymization done by the NLP team - to be described in the following sections - is removed. Then, it is ready to be sent back to the clients, according to their specifications.

In the two following pipelines, the same steps occur, differing only in terms of translation and product delivery speed and level of precision in the translation through human PE. While the *Hybrid* pipeline is asynchronous, combining machine translation with human PE, the *Realtime* pipeline, that can be seen in Figure 2, is, according to Brougham (2022), a synchronous translation based only on machine translation, without any human intervention from the human PE. It means the delivery time to the client is shorter, although the level of precision in the translation may not be equivalent to human PE.

**Figure 2 - Realtime Translation Pipeline**

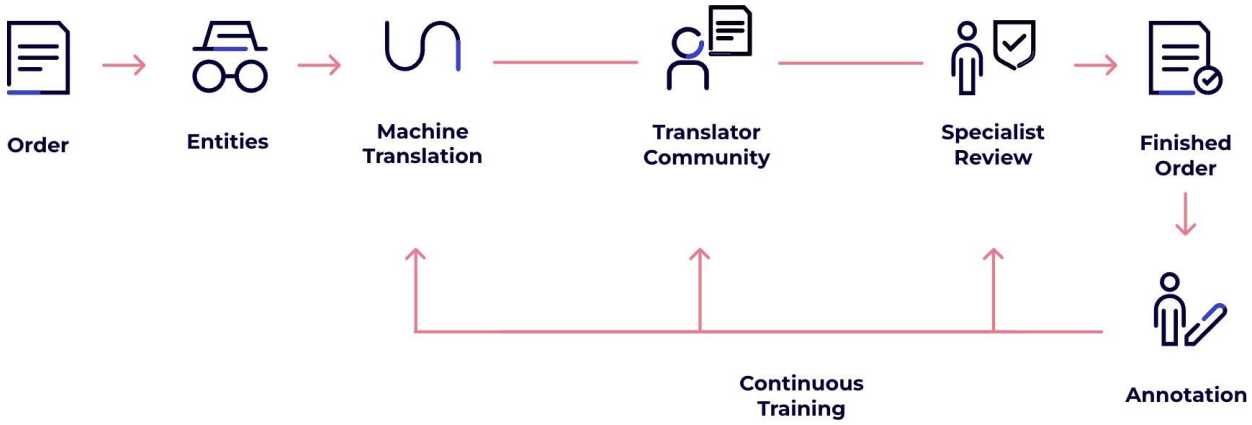



Source: Usage authorised by Unbabel, content presented by Phill Brougham.

Once translations are delivered through the pipelines, it is important to consider that the choice of the pipeline by the clients also defines the cost of the final product. In other words, if a client chooses a *Realtime* base translation, supported only by machine translation, the cost is going to be much lower than a translation done in the *Specialist* pipeline, as displayed in **Figure 3**.

In terms of the *Specialist* pipeline, which can be seen in **Figure 3**, it is an asynchronous translation as well as the *Hybrid* one. Nevertheless, it has a greater degree of human input in the process of PE. It means that depending on the type of translation chosen by the clients, it might require domain expertise in two of the steps: Translator Community and Specialist Review, which, according to Moniz et al. (2020), is a senior editor with more experience and is able to provide consistency and revision to the translation.

**Figure 3 - Specialist Translation Pipeline**




Source: Usage authorised by Unbabel, content presented by Phill Brougham.

Although there are three different translation pipelines, we must consider that some steps are present in all of them. One of those steps is the **Annotation**, done after the translation process is finished and delivered to the client. According to Paulo (2022: 13), "this process consists of identifying and labelling errors according to a specific typology, and is extremely important as it allows the retraining of the artificial intelligence (AI) systems, namely the MT, NER and QE". In other words, professional linguists will assess the text in order to guarantee not only quality control but also as a way to improve the revision of the content and the MT quality. Moreover, this assessment performed by professional linguists might help the improvement of the quality of all the other steps in the pipeline.

When necessary, the MT output is post-edited and annotated by the aforementioned Translator Community so as to guarantee high quality and continuous improvement of the MT. Within it, "there are also several automatic processes in place to predict translation quality and improve the content that is delivered to the end-users" (Silva, 2022: 24). The annotated data serve as benchmark datasets for continuous assessment and monitoring of the delivered quality. This workflow for ensuring translation quality is integral to the company's commitment to product excellence in their decision-making process.

By using the method of collecting and annotating data, Unbabel trains and improves the quality of the MT. As stated before, this annotation is done by human editors through a

proprietary tool, called Annotation Tool[3]. Throughout the annotation process, a linguist employs Unbabel's Annotation Tool to meticulously assess a translated text, identifying errors, and categorising them based on their type and attributing a severity score. The errors are annotated and classified in translated texts using the Multidimensional Quality Metrics (MQM) framework. It is a set of metrics used to evaluate the quality of translations in general and, more specifically, machine translation in the core of Unbabel's works. Developed by Lommel (2014) in the scope of the *QTLaunchPad*[4], MQM is regarded as a benchmark for manual quality assessment in the translation field. It is also important to add that MQM was designed to be adapted to its user's necessities in terms of error types and their attributed severity.

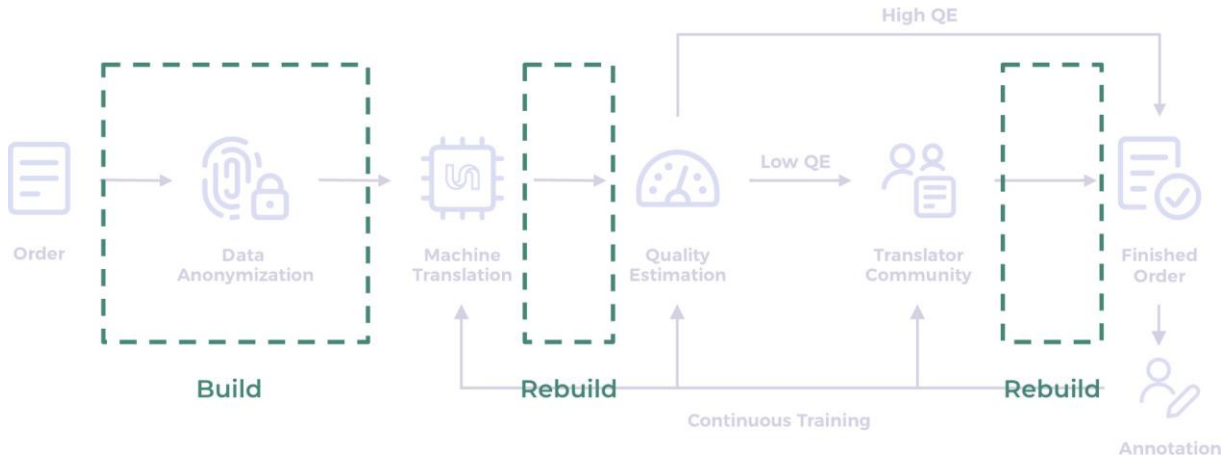
Unbabel uses a proprietary tool for annotation containing a specific error typology developed from the MQM framework. As it has been already stated, MQM was created to be an adaptable framework through which an annotator can define what are the important error types to be used in the annotation process. In that sense, Unbabel Error Typology consists "of three main categories derived from MQM's core (*accuracy*, *fluency* and *style*) that branch out into sixteen parent tags, thirty daughter tags and ten granddaughter tags" (Silva, 2022: 26). This way, not only quality will be provided but also consistency within annotations.

## 2.3. NATURAL LANGUAGE PROCESSING PRODUCTS

The internship was conducted alongside the Natural Language Processing team, also known as NLP. NLP is a multidisciplinary team composed of linguists, engineers, and artificial intelligence (AI) experts. The main goal is to understand how the human language works through the combination of computer engineering and artificial intelligence. In that sense, the NLP team aims at enhancing NLP algorithm-based systems in the shortest period of time possible.

[3] An in-house tool. More information available at: <https://annotation.tools.unbabel.com/>.
[4] https://themqm.org/mqm-history/

**Figure 4 - Natural Language Processing Team**




Source: Usage authorised by AI-Tech NLP team.

Considering the translation pipelines previously described, the NLP team works in two main stages: *Build*, before the MT stage; and *Rebuild*, after the MT stage. As it can be seen in Figure 4, what is called *Build* stage represents the pre-processing stage, before the document is sent to the MT. All the anonymization described in the pipelines is done in this stage. The *Rebuild* stage happens in two different moments: 1) immediately after the MT; and 2) before the final document is sent back to the client, when the document is restored to its original format.

### 2.3.1. NAMED ENTITY RECOGNITION (NER)

Within the NLP team, one of the most important tasks concerns NER. In that respect, NE processing is of crucial importance since it plays a significant role in a sentence and its possible errors may influence the entire quality and fluency of the output of a translation. According to Mota et al. (2022: 141), “NEs are linguistically complex structures that can occur in ambiguous contexts”, and should be flexible and versatile, so that they can be adapted and applied to different domains. Consequently, the identification of NE by NER systems is valuable, because it prevents several machine translation errors, such as hallucinations or mistranslations, for instance.

Accordingly, Named Entity Recognition is the tool through which NEs are identified. NER systems were first used in the early 80s, when the United States and the Soviet Union were

still in the context of the Cold War (Nouvel, 2016). The US Defence idea was to invest in "automatic understanding of documents, in other words, to extract essential information in extended *corpora*, granting a strategic advantage" (Menezes, 2021: 2). In a more practical perspective, NER systems enable the identification of NEs by categorising them according to their surroundings.

From a general point of view, once a NE is identified by NER systems it is "automatically blocked for translation or automatically annotated as NE of interest for further processes such as localization" (Mota et al., 2022: p. 211). NE recognition involves two main steps: 1) identifying the word or string of words; and 2) categorising the identified words within pre-set categories.

More precisely, step one involves identifying all the words or phrases that can be a NE in the source text. Although NEs can vary in domain specificity, they are also unlimited in terms of their nature, which means that new NEs may appear every day, depending on the domain. In order to identify the Personal Identifiable Information (PII), Unbabel developed a system that enables NEs recognition, such as names, addresses, telephone numbers, email addresses, URLs, IP addresses, passwords, IBANs, credit card numbers and alphanumeric numbers, to name a few.

After NEs are identified, those entities are blocked for translation or, if not blocked, they are annotated to be anonymized and/or localised. Identifying and blocking NEs is an important step because it helps to avoid MT systems hallucination. Thus, step two is related to two other processes: anonymization and localization.

In the case of anonymization, it means that the annotated NE will be replaced by a generic placeholder - if it is an email, phone number and/or reference number for instance -; or by a semantic equivalent - if the annotated NE is a person's name. Aligned with GDPR[5], all the PII must be identified and removed from the text. When it comes to the localization process, it means the annotated NEs need to be localised in the target language format when it goes to the MT step. Whenever a NE is not annotated, the text will go to the MT system and the non-annotated ones will be translated as any other word, which will most certainly result in a mistranslation or in a hallucination. Both independent modules will be better described in **sections 2.3.2** and **2.3.3**.

[5] Data Protection Act 2018. Available at: <https://www.gov.uk/government/collections/data-protection-act-2018>.

#### 2.3.1.1. UNBABEL NER SYSTEM

Since we have already detailed how the annotation process is carried out at Unbabel, now it becomes mandatory to explain and demonstrate why we chose to have an in-house system, instead of using one of the open-source systems we will detail in **section 3.2.1**. **Table 1** presents all the categories the in-house NER system identifies. Not only do we have more categories than the other NER systems, but we also have subcategories indicating more detailed and specific areas. The in-house NER system was mainly built focusing on the customer support domain so that the categories come from the demands in that domain.

**Table 1 - Annotation Categories**

| | |
|---|---|
| **General** | Name |
| | Products and Services |
| | Organisation |
| | Audiovisual Title |
| | Document Title |
| | Episode/Season Number |
| | Username |
| | Email |
| | URL |
| | File-Path |
| | Artwork |
| | Event and Campaign |
| **Location** | City |
| | Country |
| | Nationality |
| | Address |
| | Region |
| | Geo-Feature |

| | |
|---|---|
| | Construction |
| **Date/Time** | Date |
| | Time |
| **Numerical** | Number |
| | Measure |
| | Percentage |
| | Currency |
| | Phone Number |
| | Alphanumeric |
| | Device-ID |
| | ID-Number |
| | Bank-Number |
| | Credit-Card |
| | Tax-Number |
| | IP-Address |
| | Password |
| **Biomedical** | Disease |
| | Chemical |
| | Medicine |
| **Food** | Dish Name |
| | Beverage |
| | Restaurant Name |
| **Gaming** | Game Title |
| | Game Feature |
| | Game Currency |

Source: Named Entity Annotation Guidelines.

Therefore, once we understand that NER systems play a crucial role in enhancing the efficiency and effectiveness of various modules within the NLP team, we are able to realise that by improving the NEEP model, we help to improve Unbabel NER system, because the NER system includes annotations of single- or multi-word expressions, and it also may involve any sequence of letters, numbers and symbols. Additionally, the NER system serves as a fundamental component in numerous NLP applications, including speech processing for both recognition and synthesis, automatic summarization, document indexation, machine translation, amongst other areas.

### 2.3.2. ANONYMIZATION

Anonymization is one of the steps performed by the NLP team and it is only possible because of the NER system, described in **section 2.3.1**. It is automatically performed in the source text so that all the PII is hidden, not only to protect information from NMT but also from PE. Moreover, NE anonymization complies with the European AI Act (Madiega, 2021) and with the GDPR[6], as previously described, which means that it follows the laws that regulate all personal information in the European Economic Area with the aim of granting individual's absolute authority over their own personal information.

All the PII entities annotated through the NER system are automatically anonymized in the source text and replaced by generic placeholders, in the case of ADDRESS, IBAN, CREDIT-CARD, IP-ADDRESS, ALPHANUMERIC-ID, EMAIL, URL, PHONE NUMBER and DEVICE-ID; and by a semantic equivalent, in the case of PERSON NAME. A semantic equivalent is when we replace the person names in the source text with another name, in order to guarantee that gender agreement[7] will be kept in the target language when it is necessary. It facilitates the MT and the PE processes, leading to a better understanding and readability of the translated text.

If we take the European Portuguese language as an example, we have names such as Maria, Ana, Isabel or Sandra for female and João, Manuel, José or Carlos for male, to replace the original names in the source text. Whenever we have a language with gender differences in the names, it guarantees gender agreement in the sentences output.

[6] Data Protection Act 2018. Available at: <https://www.gov.uk/government/collections/data-protection-act-2018>.
[7] See **Table 2**, in **section 2.3.2**.

**Table 2 - Anonymization examples**

| Named Entities - PII | Placeholder |
|---|---|
| Credit_card | creditcard-0 |
| Name | Semantic equivalent according to the gender of the name and target language |

In addition, when NEs are identified and replaced "with corresponding placeholders [it] will lead to an increase in the number of Translation Memories matches, which promotes more accurate end-translation results, while lessening, simultaneously, the need for human post-edition" (Mota et al., 2022: 218); and, consequently, the decrease of costs in the entire process.

In the final steps of the pipeline, before delivering the translated text, all the PII entities are rebuilt/reallocated to the target text. Nevertheless, it must be considered that PII may vary depending on the purposes of the text. In that sense, we can apply a "customizable anonymization", which means customers are allowed to select the categories they want to be anonymized. Taking that into consideration, there are two relevant benefits that can be pointed out: the first refers to the extension of the anonymization that can be done wherever required in the content; and the second reflects the optimization of the commitment between the translation quality and the privacy consistency necessities.

### 2.3.3. LOCALIZATION

Localization can be mainly described as the process of converting NEs into the standard formats they must follow in their corresponding target languages. We must, firstly, annotate all the NEs and, then, there are some possible paths to follow, varying according to each strategy. One of such localization strategies is "Copy-from-Source/untranslated". It means the NEs that do not need to be transformed from source to target texts are only blocked from translation. After the MT stage, they are copied back to the target text. They may be URL, EMAIL, ALPHANUMERIC-ID or IBAN, to name a few.

Another localization strategy is “Transliteration”. It is necessary when the writing system of the source text differs from the one of the target text. For instance, we have to localise all the person names from a source in Latin script into languages such as Arabic, Chinese, Greek, Japanese, Korean and Russian, which use a different language system. NEs in those languages are prone to errors in the MT step, so we must make sure they are correctly annotated to be properly localised and transliterated.

A third localization strategy is the “Parallel-Translation”. Like all the other strategies, Parallel-Translation is done before the MT stage, especially for city and country names, since they are prone to hallucination, especially when they are written in abbreviated forms. We also use dictionaries to help in that strategy.

The last localization strategy is the “Automatic formatting”. Categories such as NUMBER, MEASURE, PERCENTAGE and DATE need to follow locale conventions in the translation process and all that is done automatically. Taking the case of date and time format as an example, languages have their own rules on how to write the date and the time, hence when translating, it must be considered the target language format for it, in order to guarantee the style and the format are accurate in that specific language.

**Table 3 - Localization examples**

| Categories | Localization |
|---|---|
| Copy-from-Source/untranslated | EN: My name is **Juliana.**<br>PT: O meu nome é **Juliana.** |
| Transliteration | EN: My name is **Juliana.**<br>RU: Меня зовут Джулиана. |
| Parallel-Translation | EN: I live in **Lisbon.**<br>PT: Eu vivo em **Lisboa.** |
| Automatic formatting | EN: It’s 4pm.<br>PT: São 16h. |

The several processes just described are a way to guarantee two core aspects: localization of NEs and anticipation of potential critical errors, since those usually affect NEs.

### 2.3.4. NAMED ENTITY ERROR PREDICTION (NEEP)

Named Entity Error Prediction (NEEP) model is an automatic system that allows NE error prediction whether after the MT step, or in the PE process involving entities. Having a model that is able to predict the NE errors can reduce the dependency on human annotators, and that would reflect in a reduction of time and costs in the entire translation process.

This model was trained on datasets with abundant examples of NE. The evaluation and classification made afterwards took into consideration the performance of the model based on precision and recall. In that respect, it means that not only the volume of data makes a good training process, but also refined and well chosen data, in terms of NE errors categories, help to enhance the model's validation.

In addition, we must state that NEEP and NER are different tools. They are carried out in different stages of the pipeline, with the NER system being used to identify and deal with the NE in the source text, as explained in **section 2.3.1**. The NEEP model, on the other hand, is used after the MT stage, in the target text, based on the trained models from the annotations. Both tools combined allow to tackle NEs and avoid critical errors.

To that respect, if the NEs are not properly identified by the NER system for proper anonymization and/or localisation, there is a high chance that the machine will make mistakes in the translation. Consequently, the NEEP model is important after the MT stage, because it identifies in the target text the NE errors made by the MT. When those errors are identified by the NEEP model, we can analyse what are the most common errors, by periodical annotations, in what type of NEs and in which languages and, thus, improve the NER system in their recognition in the source text, so that we decrease the chances of MT errors.

The work of the NEEP model relates to the NER system work regarding the identification of the most error prone NEs. As we are going to discuss along chapter 3, there are several open source NER systems available in the industry, although we use an in-house system. However, to the best of our knowledge, there are no NEEP models available yet. For that reason, the work

presented here is of great importance in the NE area, since it enables the recognition of NE errors and, thus, we will be able to improve NER systems.

Summarising, if the NEEP model is able to correctly predict the NE errors, then we will be able to retrain the NER system with the NEEP model data. Both tools – NEEP and NER – work independently from each other. However, when combined through the pipeline, they can help and significantly improve each other's performance.

## 2.4. PROJECT GOALS

The main goal of this thesis is to evaluate the NEEP model and improve it with data augmentation, by extending the current NE errors dataset with more NE errors from different clients, domains and content-types. In order to carry out this study case, we will evaluate the current status of the NEEP model by identifying the error types that the model is not covering and the ones it is capturing.

Given the results of the evaluation, we will create fictional NE errors from different clients, domains and content-types. After that task, we will use the new dataset to retrain the NEEP model and evaluate the results again for another interaction. The aim is to enhance the better identification of those NE errors without having to rely on human annotations, which might be expensive and time consuming. Accordingly, we expect to be able to, firstly, catch NE errors automatically, and then, as a second gain, identify cases where the NER system is failing and improve it as well.

# 3. STATE-OF-THE-ART

## 3.1. HISTORY OF MT

Translation is not a recent invention. Not even Machine Translation (MT) is something we can properly call recent. From a philosophical and somewhat mythological perspective, it is possible to state that translation began with the Babel Tower, when the languages of the world had appeared, legend has it. From a more pragmatic point of view, although flowery, we can establish the beginning of translation when among different people, speaking different languages, from a first contact there was a necessity for communication. Although curious and interesting perspectives, they do not represent translation as we know nowadays. Translation as a professional skill only appeared later and, as a study field, even later in human history, due to the fact that only very recently science turned its eyes to machine translation as a field of interest for researchers. Most of the studies involving translation had begun in the 1960s. However, they have advanced very quickly since then and, especially, in the last few years.

If we may build a timeline perspective, in the 1930s, there were two researches being made for electromechanical devices that would work as a sort of multilingual translation dictionaries. One of them was taking place in France, by George Artsrouni, and the other in Russia, by Petr Trojanskij. As claimed by Kenny (2018), the research carried out by Trojanskij involved better developed concepts, using a "three-stage architecture in which a monolingual human operator would parse a source text using a universal scheme that could capture all possible grammatical functions of words" (Kenny, 2018: 429-430). Trojanskij work is interesting and relevant to be mentioned because, according to Kenny (2018), it is a pioneer to a wide debate on the role of bilingual humans in the machine translation process. Furthermore, Trojanskij also advanced the idea of a three-stage MT architecture that would be a state of the art in MT by the end of the 20th century.

In the following years, especially due to the end of the Second World War, much effort was made in automated translation, as an addition to the introduction of the first computers. In that regard, the Warren Weaver Memorandum, in 1949, is one of the main pieces in the puzzle that is MT history. The Memorandum brought Weaver's ideas on how to overcome the ambiguity problems that were happening to the word by word level of the MT translation. Being a

mathematician and director of the Natural Science Division at the Rockefeller Foundation, Weaver suggested the use of statistics in order to reduce semantic ambiguities, taking the context surrounding the words into consideration.

Moreover, the Warren Weaver Memorandum "outlined the prospects and suggested various methods: the use of wartime cryptography techniques, statistical methods, Shannon's information theory, and exploration of the underlying logic and universal features of language, 'the common base of human communication'" (Hutchins, 1995: 433). All that led to a time of vast investigation in automated translation, in many different universities in the USA, coming to a public demonstration of machine translation in 1954, carried out by a collaboration between Léon Dostert from the University of Georgetown and IBM. The demonstration consisted of translating a given number of sentences from Russian into English and, although it had a small scientific relevance, it was able to cause an impression.

The following decade was of high expectation on MT advances. According to Hutchins (1995), research was being made in both senses: to improve basic hardware and to design programming tools appropriate to language processing. However, in 1966 the Automatic Language Processing Advisory Committee (ALPAC) came to change the scenery. As complex issues arose on the linguistics side, the main argument was that computers would not be able to deal with ambiguities or words that required worldly knowledge in the translation process. The ALPAC report aimed at examining the MT prospects for future investment and ended up concluding that machine translation was more expensive than human translation. Though the report might be considered biased and short sighted, as Hutchins (1995: 436) states, "it brought a virtual end to MT research in the USA for over a decade and MT was for many years perceived as a complete failure".

The scenery began to change again by the end of the 1960s. Two different projects, one in Canada and one in Europe, set a new light upon the interest in MT due to the bilingualism in the first and multilingualism in the European community. Thus, Canada invested in research to create a syntactic transfer system between English and French; and the MT main area of focus was the translation of weather forecasts. According to Kenny (2018: 432), "the Météo system stands as one of the most successful MT implementations of the twentieth century, from the point of view of its longevity but also from the point of view of its productivity", because the Météo

system was in service between 1977 and 2002, having translated over 45,000 words a day by the end of its service.

On the side of the European Communities, Systran was the system created in 1976. Back then, before the European Union existed, the Commission of the European Communities (CEC) started using Systran mainly for the language pair English-French and, afterwards, English-Russian, for military and Space Race purposes. Throughout the years other language pairs were incorporated into the system, and the European Union built another in-house system which is still being used nowadays. Systran, for now, "continues to provide MT services to other major clients using technology that has changed considerably since it was first developed in the 1960s" (Kenny, 2018: 432).

The two last decades of the twentieth century were of great development and investment in the MT field, especially due to the amplified commercialization of computers. Research on MT paradigms increased enormously, whether in rule-based systems or in corpora-based ones, as we are going to see ahead in more details. Moreover, the investment in different types of MT systems became wider as the 20th century was coming to an end. We could not only find more variety in terms of languages, but also across domains, tackling as many subjects as possible according to the advances.

Through this brief section, the aim was to highlight the most significant events that led the MT evolution across time and space. More specifically, the aim was to bring light to MT history as a way to localise the readers of this work to the approaches we are reaching at the present moment. For that purpose, the next subsection discusses the MT paradigms from its beginning up to the present state of the art.

### 3.1.1. THE MT PARADIGMS

When looking at translation, from a general point of view, as a field of work and research, a proper translation requires not only words that go from source to target language, but also context, in order to guarantee we are delivering the message from one language to another, with proper culture elements and localizations. Therefore, according to Kenny (2022: 24), translating means "focusing on texts rather than languages, [which] keeps things real, and manageable". If in the beginning translation was performed with paper and pen, nowadays,

technology advances and its fast improvements has provided translators a wide range of tools to help with everyday work. We can state that it goes from basic use of the typing machines and the computers, until the spread of the Internet and new technologies, such as Computer Assisted Translation (CAT) tools, Translation Memories (TM) and automated translation systems.

In machine translation, as well as in human translation, errors might happen. These errors may vary from trivial to serious ones, depending on the translation area. Considering that there is much research being made with the aim of estimating the quality and evaluating the outputs of the machine translation, in order to improve the performance of the automatic translation systems, works of pre and post edition of the texts can be carried out.

Many of the problems that would be easily solved by human translators may become a real challenge to the machine. For instance, we can mention ambiguities, non-isomorphism and idiomatic expressions. A human translator is able to research at any time s/he faces a problem either in the source text or in the target one. A translation performed by a machine, on the other hand, needs to be trained and retrained with a certain amount of data in order to learn the information within a language. As we are going to see in the Methodology and Results sections, our aim in this work is to improve the NEEP model in the prediction of errors performed by machine translations.

From a general perspective, machine translation can be understood as the automatic production of a target text, based on a given source text. In historical terms, machine translation "emerged in the aftermath of the Second World War" (Kenny, 2022: 32). In the years of the Cold War, there was already research being made on automatic translation, although resources were limited and scarce. As stated by Kenny (2022: 32), "automatic translation systems were applied primarily in defence, government and international organisations by the late 1960s and 1970s, and by the end of the century their use was expanding in commercial settings". In that sense, the automated translation systems are in constant evolution and improvement, and the paradigms are quickly renewed and what was considered the latest state of the art becomes obsolete overnight.

The creation of **Rule-Based Machine Translation** (RBMT) systems can be the first milestone to that history. It uses grammatical and lexical rules as a basis for machine translation. It went through three different stages: 1) the first was a type of translation done word by word with the help of dictionaries. Because it was prone to errors, the results of the translation were of low quality and demanded a greater amount of human post-edition; 2) the second stage

determined a series of pre-editions in the source text in order to predict and prevent possible machine translation errors. The interventions in the source text were mainly made in syntactic and lexical structures; and 3) the third stage was based on the existence of a universal language, although initially it departed from the same interventions as in stage two, meaning adjustments made in the source text were mainly made in syntactic and lexical structures.

In a nutshell, the RBMT system involves giving the machine enough information and knowledge about a certain language – or pair of languages – alongside with the specific rules to deal with the language(s) so that it will learn how to translate that. Unfortunately, it is a very expensive way to build machine translation because it involves specialised linguists and there are aspects of the languages that cannot be predicted. Although "rule-based systems are still being developed, mostly for low-resource languages" (Steingrímsson, 2023: 8), by the end of the 20$^{th}$ century, researchers understood the RBMT system was not practical and had little perspective for future advancements.

That was the moment Corpora-based systems started gaining popularity among researchers and translators. Also known as **Data-driven Machine Translation**, corpora were created based on a source text and its translation made by a human translator. When submitting a source text to a machine translation, it first splits the text into segments – or sentences – and, then, searches for equivalent segments, already translated, in its sort of library; the translation would be made instantly and automatically. For the segments with no equivalents, the system would, from its own library, recombine segments in order to provide a possible translation. Within the Data-driven Machine Translation we have two different types of MT: the **Statistical Machine Translation** (SMT) and the **Neural Machine Translation** (NMT).

SMT systems decide the best translation for each segment of a text from a dataset created having a source text and a human translation as basis for it. The more data the SMT systems have, the better the quality of the translation is going to be, because "the possibility that the SMT system had to continuously increment its knowledge with its translated material ensured a constant increase in translation quality" (Menezes, 2021: 21). Unfortunately for the SMT systems, the contrary is also true and the less information they have, the bigger are the chances of MT errors.

There are two types within the SMT systems: the translation model and the language model. The difference between the two models is that "the translation model is supposed to

capture knowledge about how individual words and *n-grams* are likely to be translated into the target language, while the language model tells you what is likely to occur in the target language in the first place" (Kenny, 2022: 37). Although systems based on SMT were the state of the art for over ten years, they started declining in 2015, when another approach was introduced and started gaining popularity because of its better performance results. Thus the NMT system arose. Like the SMT systems, NMT systems also use corpora composed of segments of translated texts. Nevertheless, they differ in their data processing methods. The main intention of a NMT system is to mimic the human brain's way of processing information. In simple terms, what the system does is artificially process the information as a human brain in the sense of neural connections, recognizing patterns and solving problems in the Artificial Intelligence field. According to Koehn (2017: 6), "a neural network is a machine learning technique that takes a number of inputs and predicts outputs", to which artificial neurons are linked to one another and thus they are associated with a certain weight and threshold. Moreover, unlike the SMT systems, the NMT systems are able to deal well with low-resource language pairs.

From a more detailed perspective, the NMT systems outperformed the SMT systems because of the type of models they learn information from. According to Kenny (2018: 436), training the NMT systems "mean learning the weights that will result in those distributed representations that can best ensure that the network, when called upon to translate, outputs translations that are as close as possible to the 'gold standard' human translations found in the training data". Though very similar in terms of statistical prediction, SMT and NMT systems differ in the dealing of sentences, because the latter does not need to work with *n-grams*, being able to process full sentences.

NMT systems continue to be widely used, even though there is a new paradigm arising which promises to be the next state of the art for machine translation. According to Chen et. al. (2023: 1) the **Large Language Models** (LLMs) are able to "achieve reasonable performance on traditional language tasks such as reading comprehension, translation, and summarization". Moreover, Gao et. al. (2023: 1) stated that "in recent years, there has been a growing trend towards incorporating large-scale pre-trained language models for natural language processing"; which means that the LLM models are usually trained on enormous amount of data enabling them to present great results both in language comprehension and language output.

One of the most famous examples we could cite in the work is the recently released ChatGPT[8], which, as stated by Gao et. al. (2023: 1), "is a powerful pre-trained language model developed by OpenAI, [and] has exhibited an impressive level of performance in both natural language understanding and natural language generation, as evidenced by its ability to comprehend and generate human-like responses in a wide range of contexts".

Much research has been made since ChatGPT became well-known to the general public and, although Gao et. al. (2023) demonstrated in their work that when compared to other commercial translation systems, such as Google Translate and DeepL, ChatGPT has an outperforming result when given the adequate prompts, Savenkov (2023) from Intento[9], demonstrated how it still needs improvements in domain-specific texts, such as healthcare, for instance, and for a wide range of language pairs. Altogether, the LLMs did not come as a surprise, even though their use is being systematised in order to be well adapted to companies necessities and clients demands.

### 3.2. NER SYSTEMS

Contained within the scope of the NLP work, the NER systems play an important role in the machine translation field, since they enable the recognition of NE in different areas and domains, as necessary. As stated by Menezes (2021: 24), "the dichotomy, domain-general versus domain-specific Named Entity Recognition, e.g. domain adaptation, has been a subject of particular interest for many scholars working in the field". For that reason, NER systems are in constant improvement in order to decrease the dependency on human intervention in the post edition step of the translation process.

NER systems started being used in the journalistic text domain, mainly in English. According to Agarwal et. al. (2021), those systems gained high visibility and achievement in conferences such as MUC (Message Understanding Conference[10]) and CoNLL-2003 (Conference on Computational Natural Language Learning) for instance. In both conferences, datasets were tested in NER systems, to prove their applicability and performance. Nevertheless, much work is continually being done to improve even further the NER systems outreach as well

[8] Available at: <https://openai.com/blog/chatgpt>.
[9] Available at: <https://blog.inten.to/gpt-3-translation-capabilities-8a9290731a45>.
[10] There have been seven conferences in total between the years 1987 and 1997.

as all the work involving NE. Although, as we have already stated, English was, and still is, a major language in the NER systems work, as research progressed and other countries got involved, nowadays there are far more languages also in the scope of the NER work.

Now, taking a step back to understand the development and the importance of the NER systems, we can highlight three main criteria that are supposed to be followed in order to guarantee a well-performing system: 1) high quality when avoiding too many errors; 2) extensiveness, for it must include as many NE as possible and; 3) sufficiently robustness in terms of being able to deal with a wide variety of NE. Accordingly, Agarwal et. al. (2021: 117) affirmed that "NER systems recognize entities seen in training more accurately than entities that were not present in the training set". It means that the more data the systems receive in varied domains, the better they become in recognizing the named entities.

Nevertheless, Agarwal et. al. (2021) also describes that the systems' performance decreases enormously when handling with entities never seen before in the training datasets or, even, when NEs appear within different surroundings in datasets already used in training phases. It is important to highlight that recent work in NLP relies on neural representations to expand the ability of the NER systems to learn context and names altogether. In other words, the systems have been improving to guarantee that they understand the surrounding and the context in which named entities might appear.

> "In general terms, multiple features or clues may be used in named entity recognition, but no subset of these properties is sufficient to recognize a named entity class. As we will see, features must therefore be combined in order to determine which segments of a text genuinely constitute named entities. Some of these features are more significant than others, notably those relating to the morphology of named entities (for example, the presence of capital letters), and those based on the use of dictionaries or lexicons of named entities. Finally, contextual elements may be used to support the analytical process, allowing us to take account of the likelihood of a result for a given text" (Nouvel et. al., 2016: 77-78).

Accordingly, we could state that named entity recognition can be divided into two main tasks, correlated with one another: 1) *detection* of the NE and; 2) *classification* of the NE. The detection of NEs must consider the delimitations indicating where a given NE starts and where it

finished. Subsequently, the classification of the detected NEs must follow a predefined list of categories, depending on each companies domains, specifications and necessities.

As NER systems research progressed both in the academia and in the industry fields, more domains became part of the NER framework. From journalism on, there have been advances in the healthcare domain and, in the specific case of Unbabel, we use an in-house NER system, due to the fact that we have different NE categories to address and the focus on the customer support domain, as detailed in the previous chapter.

Since their beginning, the NER systems have gone through different types of approaches when considering the recognition of the NE, in the different types of areas and domains. We could even affirm that the NER systems went from the recognition of NEs based on rules and templates, passing to the recognition based on machine learning until the present moment that the recognition is mainly based on deep learning.

At first, NER systems were trained using unannotated datasets. In other words, the systems were supposed to learn what words were NE from the context and the surroundings of the words (Agarwal et. al. 2021). By the time of the CoNLL-2003, that was considered the state of the art. However, more mature work done thereafter led to Wang et al. (2019) M-BERT (Multilingual Bidirectional Encoder Representations from Transformers) proposal. The use of M-BERT represented a new horizon in the state of the art, since, according to Wang et al. (2019), it represented a potential solution for cross-lingual transfer, even in the absence of cross-lingual objectives and aligned data. Moreover, the results demonstrated that M-BERT exhibited strong generalisation capabilities across languages in various downstream tasks, including NE recognition and Parts of Speech tagging (POS).

Accordingly, Agarwal et. al. (2021) demonstrated in their research, how NEs identified in contextualised word embeddings help solve the ambiguity that certain NEs may present, such as a proper name that can be given to a person and also be the name of a state or country[11]. Looking at more recent research in the scope of NER systems, we might find the ERNIE-BiGRU-CRF model proposed by Zhang Xiao et al. (2020) and the BERT-BiLSTM-CRF model proposed by Shen Tongping et al. (2022) which combined with a so called *attention mechanism* and the recurrent neural networks (RNN) makes "the neural network extracts sentence features and uses

[11] We could cite as examples the names "Orlando" and "Virgínia", which are common names in Portuguese and, at the same time, they are names of cities in the United State of America.

the attention mechanism to solve the problem of long-distance dependency, which effectively improves the overall [NE] recognition" (Liu et al. 2022: 67). It means that the NLP field is in constant expansion, providing great development in the NER systems research.

### 3.2.1. OPEN-SOURCE NER SYSTEMS

It is also imperative to consider the availability of NER systems, either for industry purposes or for academic ones. Although at Unbabel we use an in-house NER system, we can also cite some well-known open-source systems, such as SpaCy[12], Stanza[13] and Google Cloud Natural Language[14].

SpaCy, according to its own website, is a "pipeline component for named entity recognition". To be more specific, SpaCy is a tool developed in an open-source code trained to work with the following categories: *Person*, *Organization*, *Location*, *Date*, *Geo-Political Named Entity*, *Law*, to name a few. Besides, it is possible to use it in Python and Cython and it supports over 70 languages.

Stanza on its part, although also being an open-source code for named entity recognition, was developed in Stanford University (California, USA) and aims at achieving more academic purposes. Among the categories it is able to work with, we can cite *Person*, *Organization*, *Date and Time*, *Location* and *Measurement* for instance. Not only they provide a NER system, but also a NLP pipeline and, as Menezes (2021: 30) stated, the products are "available freely for the common user, and available for the industry through a commercial licence".

Another NER system that can be pointed out is Google Cloud Natural Language API. It addresses the categories *Date and Time*, *Credit-Card Number*, *E-mail Address*, *Flight-Number*, *Address*, among others. In addition, being a profit-oriented purpose, it is available for a varied number of languages and it is considered easy to use, even with little machine learning expertise. According to its own website, this system is able to recognise and analyse not only entities, but also sentiment and entity sentiment, handle syntax analysis and content classification.

[12] Available at: <https://spacy.io/api/entityrecognizer>.
[13] Available at: <https://stanfordnlp.github.io/CoreNLP/>.
[14] Available at: <https://cloud.google.com/natural-language#natural-language-api-demo>.

## 3.3. QUALITY ASSESSMENT PROCESSES

After taking a brief look at the MT history and the NER system history and multiple possibilities and functionalities within our work, it is also important to consider some relevant aspects of the quality assessment, in order to understand how we evaluate the output we receive after testing the NEEP model. As previously stated, we mainly focus on the *Multidimensional Quality Metrics* framework, as a way to guide the evaluation process. Nevertheless, it is interesting to take a small step back and consider some aspects of the quality assessment development, both in the manual and in automatic perspectives.

### 3.3.1. MANUAL QUALITY METRICS

The search for an evaluation of the translation quality – whether they are human translations or machine translation – is not a new topic in the translation field. Since the end of the 1980s and, mostly, throughout the 1990s and 2000s, up to the present day, there have been many attempts to create metrics that would be able to help in the process of the translation quality evaluation, focusing on systematising and boosting the process. According to Moorkens *et al.* (2018), in the beginning, the evaluation of the translation quality was performed informally and did not follow well defined parameters or tools to help in that process. Consequently, one of the biggest problems was the inconsistency of the evaluations, since it used to vary from translator to translator. In order to solve this problem, innumerable researches were made aiming the synthesis of the most common error types, so that the translators would be able to use them as a guideline.

One of the attempts to solve the inconsistency problem was to create universal metrics that could be applied to any type of text. As an example, we can cite the group of metrics created in the 1990s called *SAE J2450* and *LISA QA MODEL* (Moorkens *et al.* 2018). More specifically, *SAE J2450* is a group of metrics that consists of statistical tools to identify errors in the translation of automotive translations. It has only seven error categories[15] and two levels of severity for them. *LISA QA MODEL*, on the other hand, was developed aiming at localization

[15] According to Sirena (2004), and to give a brief dimension of the metrics we are talking about, the *SAE J2450* error types are: Wrong Term, Syntactic Error, Omission, Word Structure or Agreement Error, Misspelling, Punctuation Error, Miscellaneous Error.

and software translations. Its typologies vary between eighteen and twenty error types and they have three different levels of severity[16], depending on the error type.

It is worth mentioning that, despite those two primary efforts in creating evaluation metrics, none of them was able to answer the expectations and necessities of the translators. Even the *LISA QA MODEL*, which seemed more promising because of its richness in error type details, resulted in losing its strength by 2011. Therefore, throughout the 2000s and beginning of the following decade, much investment was dedicated to developing metrics able to evaluate the translation quality (Moorkens *et al.* 2018). As a consequence, two other groups of metrics appeared in the translation market, proposing new frameworks: 1) TAUS (*Translation Automation User Society*), developed in Amsterdam, Holland, which created the *Dynamic Quality Framework* and; 2) the project *EU-funded QTLaunchPad* (Moorkens *et al.* 2018).

Between the years 2012 and 2014, the *German Research Centre for Artificial Intelligence*, in Berlim, Germany, developed the *Multidimensional Quality Metrics*[17] – also known as MQM. According to Snow (2015), the MQM framework was developed using both SAE J2450 and the LISA QA Model. After analysing the most common error types, and taking into consideration the evaluation systems that already existed, the MQM framework established, as Moorkens *et al.* (2018) demonstrate in their research, that the most common error types were: *terminology*, *omission*, *punctuation*, *consistency*, *grammar*, *spelling*, *mistranslation* and *style*.

As mentioned before, one of the biggest problems with the analysis of the translation quality was the lack of consistency among the results obtained by the translators. In order to solve this problem and enhance the evaluation results, following the MQM framework, each translation should be evaluated according to its own aims. In other words, the MQM assumes the objective of the translation. Thus, the evaluation must be done taking into account the specific details of the text. To define the metrics, we must consider the types of problems we seek to solve and that is exactly what makes a given evaluation trustable.

In accordance with Moorkens *et al.* (2018), in 2014, the MQM gathered strength with the *Dynamic Quality Framework* (DQF), developed by TAUS. Differently from the MQMs, the DQFs were created taking into account the experiences shared by *Language Service Providers* (LSP). Additionally, when we consider that the manual quality metrics yield a quality score and

[16] The levels of severity are Critical, Major and Minor. The greater the severity of the error, the larger the assigned points, indicating its gravity (Moorkens *et al.* 2018).
[17] Available at: <https://themqm.org/error-types-2/typology/>.

facilitate an intricate classification of translation mistakes, taking the LSPs perspective helps adopting a methodical approach to assess translation quality. Therefore, the combination of the two evaluation systems turned the metrics into a more consistent system, more focused on the users' necessities.

When evaluating quality, a wide number of factors come into play, including the classification of quality problems and the method of assigning scores. To establish and structure issue types, a hierarchical system with a tree-like structure is essential within an error typology. Being highly hierarchical, MQM predicts that the error types could have up to four layers – or subcategories – of errors. It is also valid to add that not all the error categories have subcategories. Nevertheless, the categories presenting more layers help the analysis to be more accurate. "The more ramified the hierarchy, the more specific the issue is" (Gonçalves, 2021: 31). Additionally and in accordance with MQM own website[18], the metrics have been going through changes and adaptations along the years, which means there have been different versions of the metrics and lately the MQM-Core has taken over, enhancing and modernising DQF:MQM while displacing it in the process.

In the scope of the present work, we use the MQM typology focused on categories related to the NE annotations. In the previous section, we describe why and how the NE annotation works, within the in-house NER system.

Alternatively to the manual quality metrics, we have the automatic quality metrics. On the one hand, the manual quality metrics can yield more fine-grained results; on the other hand, they also demand substantial time and human resources, which means they demand a large amount of money. Consequently, and as Silva (2022) stated, the use of that type of metrics reflects on LSPs dealing with significant translation quantities. Besides, it becomes impractical to assess all translations using manual metrics and for the enhancement of systems and models, which need frequent and rapid evaluations.

### 3.3.2. AUTOMATIC QUALITY METRICS

There has been a continuous development of the automatic quality metrics and their use has gained extensive popularity, both independently and also combined with the already cited

[18] Available at: <https://themqm.org/mqm-history/>.

manual quality metrics. These metrics generate a value based on an algorithm that indicates the quality of a translation. However, they do not provide understanding into the specific errors that may be present in the translation - that is the moment the manual metrics would be helpful. According to Papineni et al. (2002), the automatic metrics outcomes enable developers of MT metrics to observe how daily modifications impact their systems. Additionally, the automatic metrics must be applicable to a wide variety of languages and domains and sensitive to minor nuances among them. Besides, they must be consistent in their scoring as well as fast, in order to keep up with the volume of work.

Among all the automatic metrics, we will briefly describe BLEU (Bilingual Evaluation Understudy), METEOR (Metric for Evaluation of Translation with Explicit Ordering), BERTScore (Bilingual Evaluation Understudy with Representations from Transformers), and COMET (Crosslingual Optimised Metric for Evaluation of Translation), which is Unbabel's in-house metric.

In the beginning, automatic metrics aimed at assessing the quality of MT and have primarily focused on evaluating the "similarity between an MT-generated hypothesis and a human-generated reference translation in the target language" (Rei et al., 2020: 2685). Conventional metrics used to focus their attention on lexical-level attributes. For instance, they assessed the frequency of overlapping n-grams between the MT hypothesis and the reference translation. Thus, BLEU was the first automatic metric that reached the market. It was developed by IBM in 2002 and "it works on a reference-based method, meaning that it calculates a quality score based on how the MT output correlates with reference translations previously produced by humans" (Silva, 2022: 35). Furthermore, BLEU takes into account various reference translations, enabling the use of diverse word choices while translating the same source word. It operates as a precision-oriented metric, calculating precision for many n-gram types and aggregating the precision scores of these distinct n-grams into a single score.

In 2005, another automatic metric would be developed: METEOR. It was mostly developed considering BLEU weaknesses. Banerjee *et al*. (2005: 1) stated that METEOR is "an automatic metric for machine translation evaluation that is based on a generalised concept of unigram matching between the machine produced translation and human-produced reference translations". In other words, METEOR works by establishing a connection between the metric score and human assessments of translation quality. It derives a score for the alignment through

the uses of these attributes: unigram precision, unigram recall, and a measure of fragmentation (Banerjee *et al*., 2005).

After the advance of NMT systems and the subsequent improvement in MT quality, metrics such as BLEU and METEOR began to prove inadequate for evaluating the quality of those translated outputs. In the following years, BERTScore became the state of the art in automatic metrics. According to Gonçalves (2021), it assesses the similarity of words through the use of contextual embeddings. Being an unsupervised method, BERTScore generates contextual representations of text sequences. These sequences are constructed by segmenting input text into word pieces, accommodating unknown words by breaking them into familiar character sequences. Zhang *et al*. (2019) affirmed that BERTScore is able to establish similarity between reference and candidate words through separate recall and precision calculations, which are performance metrics to be discussed in the following section. Nevertheless, Golçalves (2021), as well as Rei *et al*. (2020), defend that the metric's dependency on embeddings from the underlying model introduces constraints and results in frequently elevated scores and the possibility of perceiving similarity among unrelated sentences.

Another one of the latest automatic metrics is BLEURT, developed by the Google Team. Working as an assessment metric within Natural Language Generation, BLEURT generates a score by considering a reference-candidate sentence pair. Given a certain pair of sentences, it generates a score that reflects the fluency and degree to which the candidate captures the meaning of the reference. BLEURT is a metric that goes through a training process, specifically as a regression model trained with ratings data. The core of this model is built upon previously cited BERT. Moreover, Gonçalves (2021: 28) explained that "this model was trained for English by extracting grammatically diverse sentences from Wikipedia in order to be evaluated by different generalised systems".

Last but not least, we have COMET, Unbabel's automatic quality metrics. As defined by Rei *et al.* (2020: 1), COMET is a "neural framework for training multilingual machine translation evaluation models". Its approach benefits from the latest progress in cross-lingual language modelling to make prediction mimicking human evaluations, because it has been trained with DA[19], HTER[20], and MQM framework. Besides, COMET is distinct from the others

[19] Direct Assessments.
[20] Human-targeted Translation Error Rate.

due to its two architectures: the Estimator model and the Translation Ranking model. The difference between them relies on their training objectives. The Estimator model assesses a quality score, while the Translation Ranking model minimises the gap between what is considered a “the best” hypothesis and both its related reference and its source reference.

### 3.3.3. PERFORMANCE METRICS

Performance metrics are used to determine if a given system and/or model is working properly or not. They can be applied to NER and MT systems, as well as the NEEP model, which is the focus of this work. The performance of the NEEP model is assessed with the standard performance metrics “Precision”, “Recall”, “F-measure” and “Accuracy” (Makhoul *et al*., 1999), as shown in Figure 5. The results of this assessment will be further discussed in the Results chapter.

**Figure 5 - Performance Metrics**

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F\text{ - }measure = 2x \frac{Precision * Recall}{Precision + Recall}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN'}$$

Source: Makhoul et al. (1999).

Precision is calculated by the number of True Positives (TP) divided by the number of True Positives (TP) and False Positives (FP) sum. A TP means that the system identifies an existent error. In other words, it works according to how it should really do. On the other hand, a

FP hit happens when there is not an error but the system catches it as if it was an error. When that is the case, it means that the system did not work, because it is giving false alarms. Precision, in this context, indicates the percentage of NEs found in a dataset that are correct. A good system performance happens when the number of TP is higher than the number of FP, increasing the percentage of Precision.

Differently from Precision, Recall is calculated by the number of TP divided by the number of TP and FN sum. FN represents the number of misses and tends to happen when there is an error but NEEP does not catch it. Thus, by calculating the Recall, it is possible to identify the number of Named Entities that NEEP marked as an error and that were, in fact, correctly identified. When Precision and Recall are fused together the result is a harmonic mean called F-measure.

The Accuracy metric is calculated considering all the four classes of prediction. It is important to show the ratio number resulting from the TP and TN sum, divided by the sum of all of them: TP, FP, TN and FN. Furthermore, TN occurs when there are correct rejections. In that case there is no error and NEEP does not catch any of it. In other words, it means the system worked well. In that sense, the percentages of Precision and Recall of the first experiment will be shown and explained in the Results section.

# 4. METHODOLOGY

In the present chapter, we will describe the methodology used in the experiments carried out along the internship. This chapter is divided in two parts: the first one corresponds to the description of the training and the testing datasets used in the first version of the NEEP model; and the second part describes the data augmentation process, focusing on building a new dataset to retrain the NEEP model, in order to address the most common NE errors.

## 4.1. DATASETS

The first NEEP model (NEEP v1) was trained with historical data, meaning data collected and annotated along seven years according to the MQM metrics. This dataset comprises hundreds of thousands of segments containing NE errors. As previously detailed in the NEEP section (**2.3.4**), this model allows us to predict NE errors, both in the MT step or in the PE one, focusing on the target text.

Differently from the NE Annotation we described in **Chapter 2**, the error annotation is a step that takes place after we deliver the translation to the client. This process entails identifying and categorising errors using a specified typology - in this case, the MQM typology. In our case study, since we work specifically with NE, we analyse the output of the NEEP model only by classifying the NE errors. Therefore, following Unbabel's guidelines for MQM annotation, we selected the error categories "Wrong NE" and "Locale Conventions", and all the subcategories within them, so as to specify the error types we found and fine grain their analysis.

In order to ensure we are properly analysing and classifying NEs, we first need to guarantee we know what a NE is. As we have already explained in the previous sections, NEs are, for example: 1) People's names (including surnames, aliases and usernames); 2) Company, team and product names (including model specifications); 3) Country, city and all sorts of location names; 4) Email addresses and URLs; 5) Numerical entities (including currency and measurements, phone numbers, credit card numbers, passwords...); 6) Date and time expressions; and 7) Postal addresses.

When classifying a NE error as "Wrong Named Entity", we are mainly considering problems related to 1) Spelling, whitespace and/or capitalization errors; and/or 2) Any other

translation problem such as NEs being mistranslated, untranslated, unnecessarily translated, wrongly transliterated, omitted and/or added to the MT output. “Locale Conventions”, in its part, are errors that transgress locale-specific content or formatting requirements. That is an important part of this work, since different languages require different types of localization, such as “Address Format”, “Currency Format”, “Date/Time Format”, “Measurement Format”, “Number Format” and “Telephone Format”. These categories are quite often composed of numbers in their numerical written form[21], which means they are classified as “numerical entities”, hence, if not properly identified and annotated by the NER system, they might incur in error.

The error annotation becomes a fundamental part of the present work because that is how we are going to evaluate the NEEP model performance. Both the evaluation and the manual validation of the model’s performance will be conducted by two expert annotators. After testing the first version of the model, we are going to be able to analyse and annotate the NE errors the model was best at identifying and the ones it failed the most. Additionally, in the NEEP model v2, we will compare how the model improved in the error prediction.

After training, the NEEP model was tested in a different dataset, comprising 4,177 segments (see **Table 4**), with NE errors, no errors and distinct types of errors, in order to validate the predictions of the model and to better assess the recall and precision of such model.

The testing dataset was also created with data annotated by human annotators, using the MQM framework. For that testing task, we only selected data annotated as “Wrong named entity” and “Locale convention”, in order to evaluate if the NEEP model would be able to predict if those annotated entities are well annotated. It was the first task carried out in the internship.

**Table 4 - Number of segments per content type**

| DATA | NUMBER OF SEGMENTS |
|---|---|
| NE errors - Customer support | 1,394 |
| NE errors - Non-customer support | 1,067 |

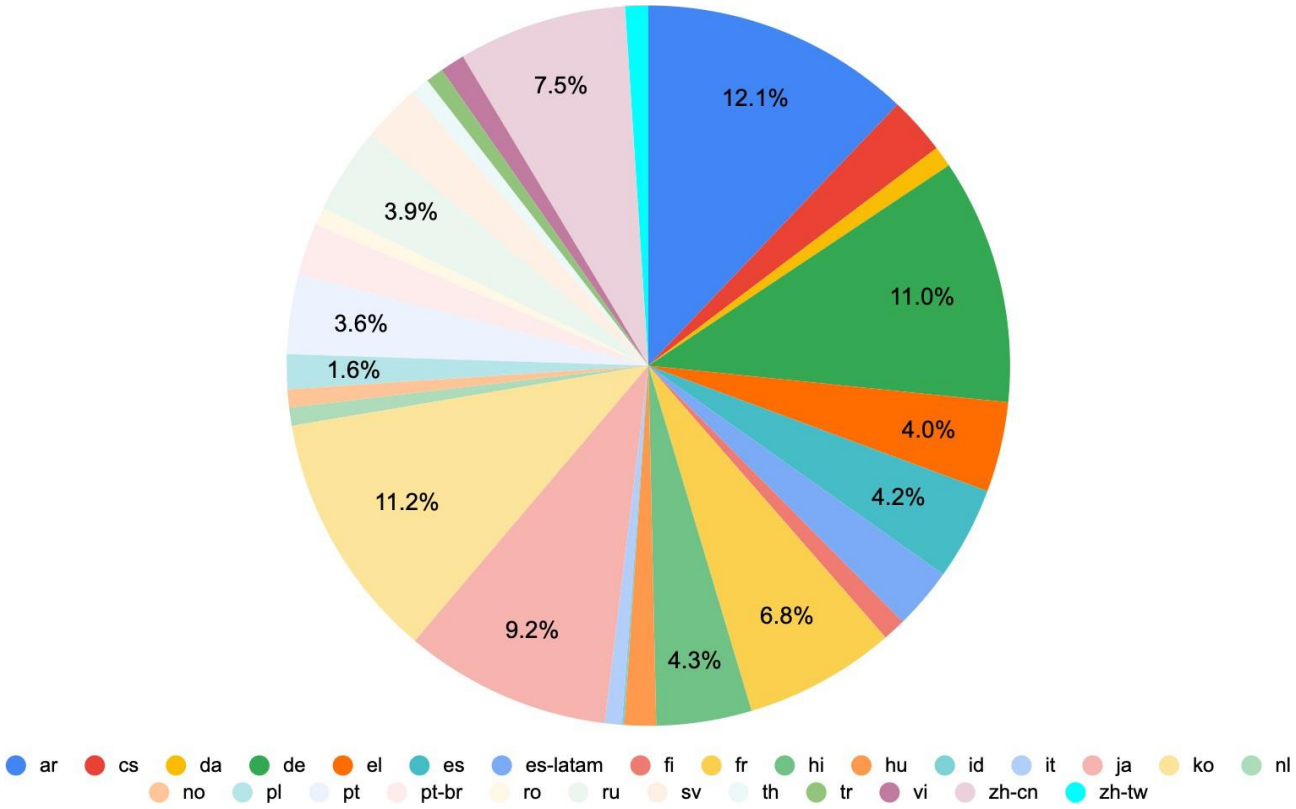
[21] Sometimes they may appear written out in full form and still be considered NEs.

| | |
|---|---|
| No errors - Customer support and Non-customer support | 850 |
| Other errors - Non-customer support | 866 |
| Grand total | 4,177 |

Source: NEEP v1 test dataset.

The testing dataset consists of segments that were extracted from different domains and can be classified in terms of content-type, as displayed in **Table 4**, into “NE errors - Customer support”, “NE errors - Non-customer support”, “No errors - Customer support and Non-customer support” and “Other errors - Non-customer support”. Among the total of segments classified as “NE errors”, there are NE errors in the MT step and in the final step, meaning they have already been edited by human translators. When it comes to “Other errors”, this content was chosen to evaluate if the model does not wrongly identify non-NE errors as such. For the ones classified as “No errors”, the main goal was to verify if the NEEP model did not catch as NE errors segments with no errors at all.

**Figure 6 - Distribution per language**




Source: NEEP v1 test dataset.

All the 4,177 segments were distributed in twenty eight different target languages, with English as the source, as it can be seen in **Figure 6**. Although the number of segments per language is unbalanced, they were chosen based on the availability of data per language. In other words, there is a low number of segments for some target languages, which can masquerade, in a certain way, the analysis for that language. This data was extracted from MQM annotations and these are performed on samples of data based on high volume of translations per clients. Therefore, there are languages with higher volumes and others with less, thus resulting in an unbalanced distribution of annotations per language pair. It will be further explained and exemplified in **chapter 5**.

## 4.2. DESCRIPTION OF THE EXPERIMENTS

### 4.2.1. NEEP ANALYSIS

In order to conduct the first task, evaluate the NEEP model performance in predicting NE errors, a fine-grained analysis to discriminate the root cause of the different NE errors was carried out based on the MQM annotations. In the testing dataset, we only selected data annotated with the following error categories: “Wrong named entity” and “Locale convention”. Within these two main categories, a series of subcategories were created to better filter the identified errors. In “Wrong named entity” the subcategories were: “spelling”, “whitespace”, “capitalization”, “mistranslation”, “untranslated”, “wrongly transliterated”, “omission” and “addition”. In “Locale conventions”, the identified errors were classified as: “address format”, “currency format”, “date/time format”, “measurement format”, “number format” and “telephone format”.

The dataset is composed of 11 different fields, containing specific information according to what is necessary to evaluate the NEEP model performance. The first field called “Target_language” is filled with the specific target language into which the segment is translated to. As stated before, it is composed of 28 different languages. The next field, called “source”, has the segment in English, the only source language used in this entire dataset. It is followed by the field called “target”, which contains the segment translated by MT.

In the field “NE category”, there is information assigned from the NER system. As it has already been stated, NER and NEEP are both systems that work independently from each other. However, we must consider that a good performance of one of them reflects in a good performance of the other. Thus, taking the segments of the dataset into consideration, when the NER system does not identify a NE in a given segment, there is a higher chance that the NE may incur in error when translated by the MT.

Additionally, it is also plausible for the NER system to miss identifying a NE or miscategorizing it. However, whenever such errors do happen, the NEEP model can still identify a possible error in a NE after the MT stage. Thus, both systems can work separately, although, when allied, they provide a better performance for each other. The next field in the dataset is “NEEP errors”. In this field we can find the NE errors identified by the NEEP model in the target

segment. Besides that, the following fields are dedicated to the analysis of the model's performance.

To analyse the NEEP dataset performance, we created distinct fields with a set of questions to be answered with "yes" or "no". The field is called "Is there a NE error?" tackles the absence/presence of a NE error in each segment of each target language. The answer being "yes", it was important to check if the NEEP model caught the NE error by answering the question in the following field called "NEEP captured the NE error?". If we mark it as "yes", the span was then checked, in the field "NEEP captured the right span?". Although it was important to consider the span, the accuracy of the *right* span was not the focus of this research.

When NEEP predicts the existence of a named entity error, it is important to analyse if the model does it correctly. Now, let's suppose that there is a named entity error in the segment and the NEEP model identifies the error correctly. The following step is to analyse if the model was able to catch the right span. In Figure 6, we have an example of a segment from the dataset. All the fields previously explained are present. Besides them, there is another field, "Which error type?" that represents the classification according to the MQM typology. After identifying and labelling the error type caused by the MT, a fine-grained analysis was made in the "Comments" field. The main goal is to highlight the errors that the model is able to detect correctly.

**Figure 7 - Dataset Example**

| Target Language | Source | Target | NE category | NE Errors | Is there a NE error? | NEEP captured the NE error? | NEEP captured the right span | Which error type | Comments |
|---|---|---|---|---|---|---|---|---|---|
| hi | (2020, October 21). | (2020, October 21). | ['DATE'] | 2020 \| , \| October \| 21 | Yes | Yes | Yes | Wrong NE | Date untranslated |

Source: NEEP v1 test dataset.

**Figure 7** also covers another scenario. Let's suppose that there is a NE in the segment but the NEEP model does not identify it. In that case, in the field "NEEP captured the NE error?" we have to select "no" and there is no span to check, so that specific field remains blank. The fields "Which error type" and "Comments" are crucial whenever NEEP does not catch NE errors, because we have to be able to identify what are the errors the model is missing. Following the

annotated error types and comments we expect to be able to identify the reason behind the failures of the model, in order to enhance it and, consequently, avoid this type of error from repeating.

**Figure 8 - Dataset Example**

| Target Language | Source | Target | NE category | NE Errors | Is there a NE error? | NEEP captured the NE error? | NEEP captured the right span | Which error type | Comments |
|---|---|---|---|---|---|---|---|---|---|
| pt | I have checked and I have found that your return has reached us on December 23, 2021. | Verifiquei e descobri que a sua devolução nos chegou a 23 de dezembro, 2021. | DATE | | Yes | No | | Wrong NE | Preposition missing |

Source: NEEP v1 test dataset.

Examining the dataset comprehensively, the analysis of the NEEP model performance is a way to recognize what are the strengths and weaknesses of the model. The “Comments” field enables us to create a pattern for the error types and the subcategories of error types. The aim is to facilitate not only the analysis of what went wrong with the model, but also create strategies to enhance the NEEP model performance.

### 4.2.2. DATA AUGMENTATION

In order to retrain the NEEP model, we collected annotated data from the NEETS (Named Entity Error per Thousand Segment), in the MT and in the final steps of the pipeline. The data was collected within 6 months, from the customers. The main purpose was to provoke errors in the NE, so that, when retraining the NEEP model it would become more aware of the NE errors, especially in the categories it used to fail the most.

To create those errors we used *smaug*[22], a package for multilingual data augmentation that offers alterations focused on changing specific aspects of sentences, such as NEs, which are the focus of this work. In a more pragmatic sense, the script automatically changes the NEs from the source input by other NEs, giving an output of perturbations that we are going to use to retrain the NEEP model. When using the script, we are not able to specify the NE categories we are feeding it with. More interestingly is the fact that *smaug* uses the NER open-source system provided by the Stanford University (Stanza), which has fewer categories than Unbabel's NER system. Therefore, we used the script with all the possible categories, following Stanza's availability and afterwards, to overcome its limitation, we manually added errors that were not being considered by the *smaug* script.

To build the dataset for augmentation, we firstly separated the dataset by two types: the first has NEs in general and the second is focused on errors in numbers related to NE, such as date, time and currency, for instance. In the MT stage, we had available data in the following languages: ar, de, el, es, es-latam, fr, it, ko, nl, pt-br, ru, sv, th, vi, zn-cn; and in the final stage we had available data in the following languages: ar, de, el, es, es-latam, fr, it, ko, zn-cn.

When using *smaug*, we observed that data augmentation did not work for a few languages, which may be related again to the fact that some NE categories were not correctly identified in the source. For these languages, we also tried to perform the data augmentation manually, taking into account the constraints of the amount of languages we speak. For the numerical categories, it was quite simple to generate errors, but the same did not occur for general NE categories, such as person names or products and organisations. Thus, we obtained a rather unbalanced dataset of augmented data, as shown in **Table 5**.

Given the information in **Table 5**, we highlight that the script provided errors in the following languages: ar, de, es-latam, ru, th and zh-cn. For all the other languages appearing in the table, we manually generated the errors, by changing one NE for another, following the scripts pattern.

**Table 5 - Number of segments per language**

| Language pair | Number of segments |
|---|---|
| en-ar | 27 |

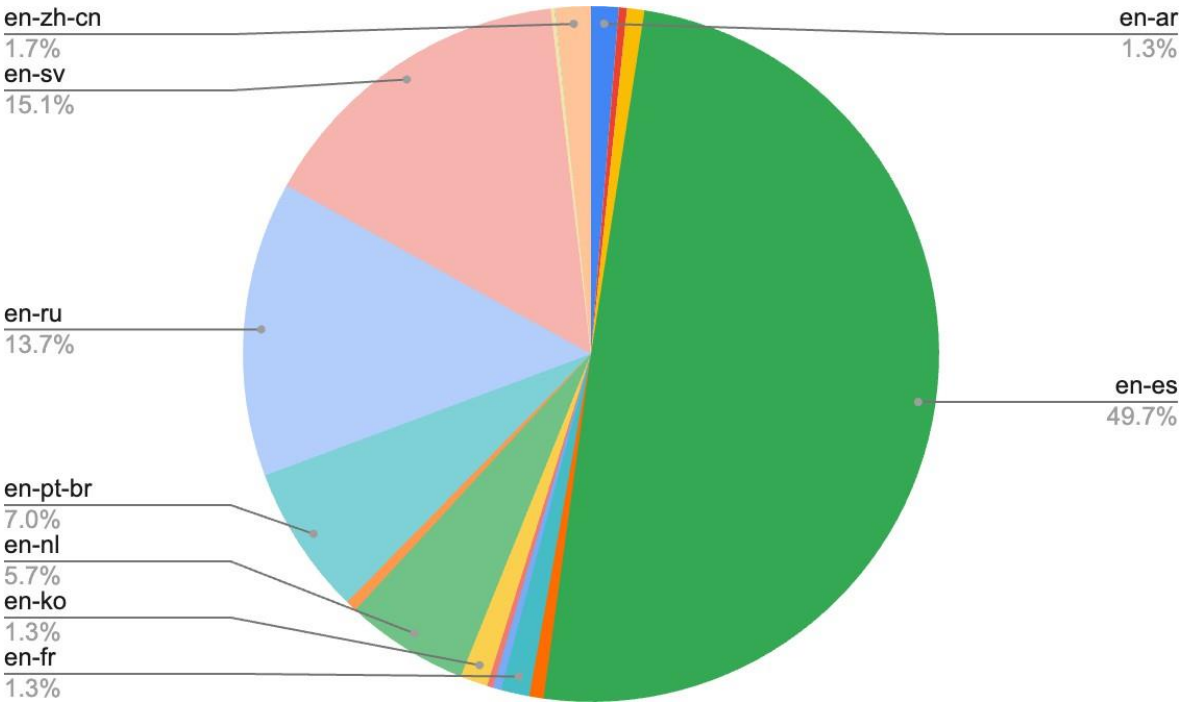
[22] Available at: <https://pypi.org/project/unbabel-smaug/#description>.

| | |
|---|---|
| en-de | 8 |
| en-el | 17 |
| en-es | 1050 |
| en-es-latam | 14 |
| en-fr | 28 |
| en-it | 8 |
| en-ja | 6 |
| en-ko | 27 |
| en-nl | 121 |
| en-pl | 11 |
| en-pt-br | 147 |
| en-ru | 290 |
| en-sv | 319 |
| en-th | 3 |
| en-vi | 1 |
| en-zh-cn | 35 |
| **Grand Total** | **2112** |

Source: Augmentation dataset.

In **Figure 9** we can see the volume of data distributed by language. It is imperative to add that we are aware that there is an inconsistency in the number of segments per language due to the available data in the NEETS dataset and to the amount of data we were able to create with the *smaug* script and manually. For that reason Spanish stands out in the figure, representing almost half of the data we used to build the augmentation dataset.

**Figure 9 - Augmented data distributed per language**




Source: Augmentation dataset.

After we have all the results from data augmentation, whether automatically or manually, we assemble them in a single dataset. This new dataset is composed of 7 different fields, containing specific information according to what is necessary to retrain the NEEP model. The first field, “Language pair” presents all the languages we were able to dispose of data to build this dataset. After that, we considered it important to define the “Register”, as formal or informal, to ensure the model retraining would be more precise. The two following fields, “Source” and “Target” present the segments in the given language pairs.

The three following fields are the most important ones. In “Perturbations”, we can see the same segment as in “Target”, but now presenting the changed NE, as displayed in **Figure 10**. To ensure we were doing a proper job with the data augmentation, all the errors generated were validated by an expert, according to the MQM framework. Thus, in the field “NE error”, we see the error category, and in “NE” we specify it, using NE annotation subcategories to precise what happened to the named entity after provoking the errors. Additionally, we maintain all the information in the dataset standardised and homogenised.

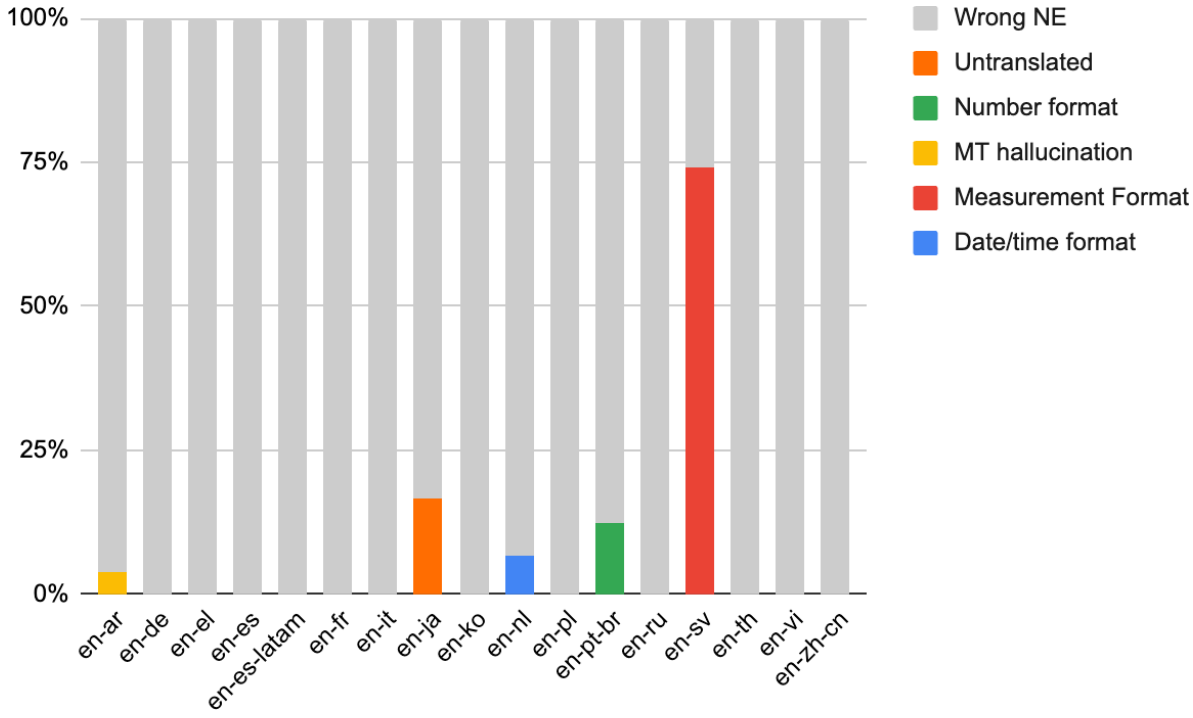
**Figure 10 - Augmentation Dataset Example**

| Language pair | Register | Source | Target | Perturbations | NE error | NE |
|---|---|---|---|---|---|---|
| en-nl | formal | Not to worry—you can still join the challenge through October 27, 11:59 pm local time. | Geen zorgen, je kunt nog steeds meedoen aan de Challenge tot 27 oktober 11:59 (23:59) uur plaatselijke tijd. | Geen zorgen, je kunt nog steeds meedoen aan de Challenge tot 27 oktober 11:59 pm uur plaatselijke tijd. | Date/time format | Wrong time localization |

Source: Augmentation dataset.

Before we could use the new dataset to retrain the NEEP model, and as a way to compensate for not having all the language pairs we needed, we established that it was important to give the model the proper NEs, and the ones it failed the most previously. Therefore, in **Figure 11**, we present the categories of NE, distributed per language, which we focused on augmenting.

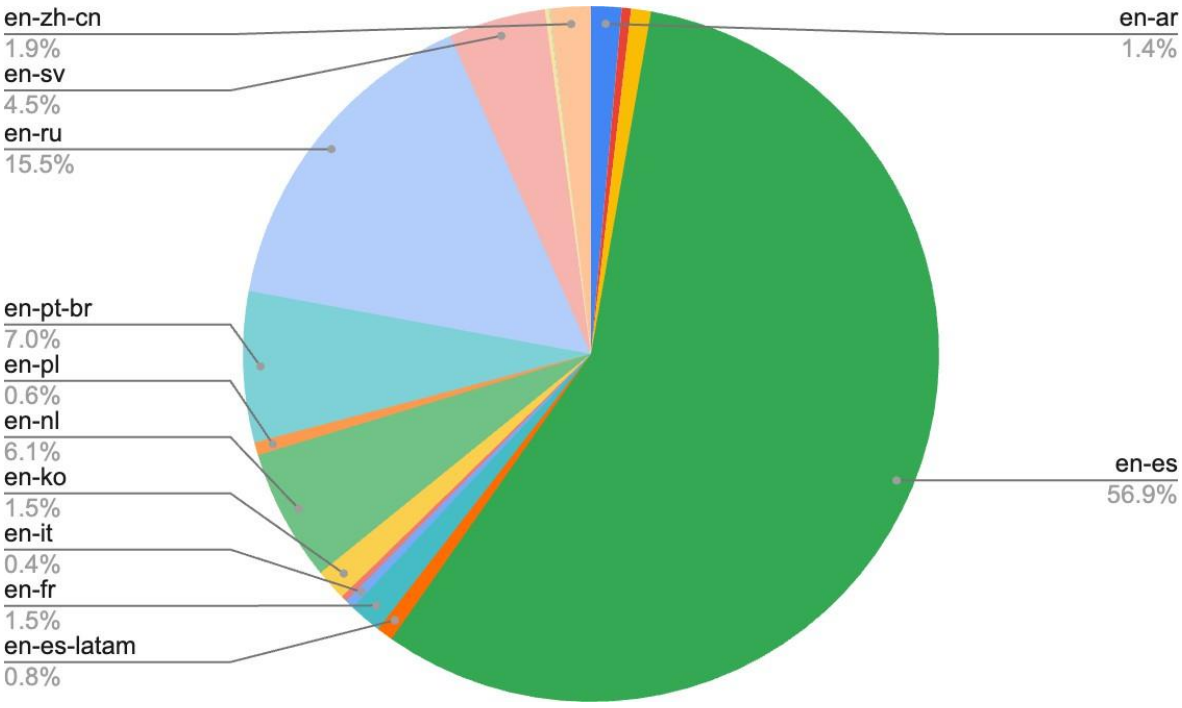
**Figure 11 - NE categories distributed per language**




Source: Augmentation dataset.

The category “Wrong NE” has the biggest amount of data, because it is the one more prone to NE errors; thus we worked on feeding the model more information on this category. Furthermore, the category “Untranslated”, which had not been used in the first NEEP model training, now has a certain amount of data. We consider it a great gain in this new training dataset.

**Figure 12 - Percentage of "Wrong NE" per language**




Source: Augmentation dataset.

In **Figure 12**, we emphasised the distribution of "Wrong NE" per language in the training dataset, as a way to ensure we have a wide variety of language pairs in that specific error type. Notwithstanding, we must reiterate once more the fact that the data inconsistency across languages has led to an uneven balance in our overall representation.

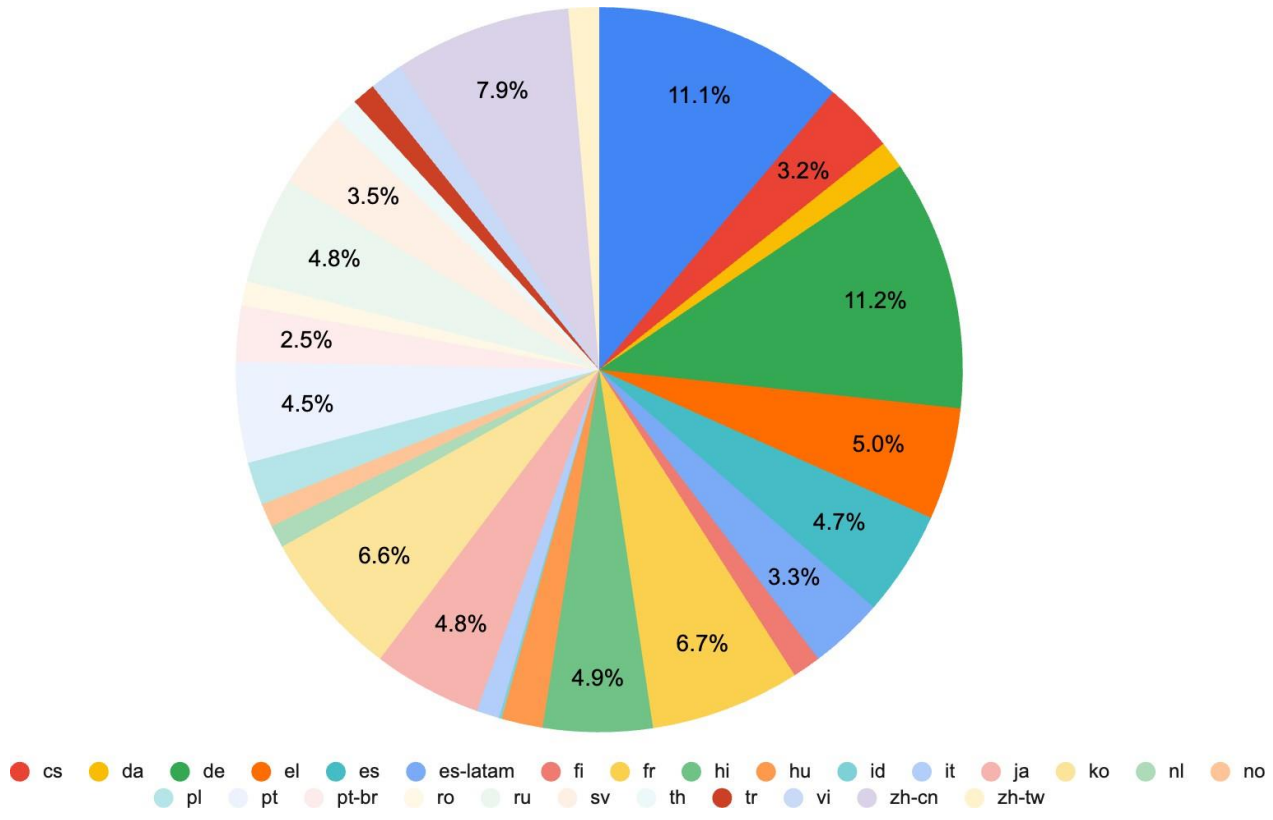
### 4.2.3. THE NEW NEEP MODEL: NEEP v2

Given the analysis of the first NEEP model performance, which results are detailed in **Chapter 5**, we realised that a few adjustments were necessary when gathering the data to retrain it. For that reason, some of the original features were altered and the new NEEP model (NEEP v2), which we are going to refer to as NEEP v2, is more focused on learning from sentences containing NE errors. Note that NEEP v1 comprises NE errors, as well as segments with other error types and segments with no errors at all (see **section 4.1**).

Comprehensively, to retrain the model, the sentences without NE errors were all removed from the training dataset. That increased the fraction of sentences with NE errors from 0.1% to 5%. This also helps the model focus better on the NE errors and be less distracted by sentences without any errors, as we previously stated. All that means that now we have a new model that is slightly different from the previous one. The following results will be analysed taking that into consideration.

In order to test and evaluate the NEEP v2 model, we selected a sample of 3,164 segments from the NEEP test dataset and ran the NEEP v2 model to be able to compare its outputs with the NEEP v1. In **Figure 13**, we have the percentage of segments per language.

**Figure 13 - Distribution per language, NEEP v2**




Source: NEEP v2 test dataset.

# 5. RESULTS AND DISCUSSION

## 5.1. OVERALL NEEP V1 RESULTS

Results for the NEEP v1 performance showed that the model performed very well not labelling as NE errors what was really not a NE error. Out of the total of 4,177 segments analysed, 1,864 did not have any error, and the model only failed labelling 3 of these segments, as displayed in **Table 6**.

On the other hand, when there is a NE error in the segments, the NEEP performance tends to drop. From a total of 2,313 segments with NE errors, the model did not catch 63% of them. Only 37% were well annotated by the model, which leads us to substantially low recall results, as we will better explain in the following sections.

**Table 6 - Amount of NE errors**

| | **NEEP v1 captured the NE error?** | | |
|---|---|---|---|
| **Is there a NE error?** | No | Yes | Grand Total |
| No | 1,861 / 99.8% | 3 / 0,2% | 1,864 |
| Yes | 1,465 / 63% | 848 / 37% | 2,313 |
| **Grand Total** | **3,326** | **851** | **4,177** |

Source: NEEP v1 test dataset.

**Table 7 - Performance measurement: NEEP v1**

| **TP** | **TN** | **FP** | **FN** | **Precision** | **Recall** | **F-measure** |
|---|---|---|---|---|---|---|
| 20.30% | 44.55% | 0.07% | 35.07% | 0.9965 | 0.3666 | 0.5360 |

Source: NEEP v1 test dataset.

Results show that, although the precision is high, reaching 0.9965, recall is very low, reaching only 0.3666. On the one hand, the high score in precision indicates that when the NEEP model identifies a NE error, it tends to do it well and is able to identify the correct error. On the other hand, the low recall score indicates that NEEP still fails to identify most of the NE errors.

This leads us to some questions such as “What are the most common errors the NEEP model is (not) identifying?” or “What are the languages in which the model fails/succeeds the most in identifying the errors?”. The answers to those questions are going to be addressed throughout the following sections.

### 5.2. ERROR DISTRIBUTION ANALYSIS PER NE CATEGORY

After analysing the dataset according to the established methodology, we were able to separate the segments containing NE errors from those not containing them. Then, focusing only on the segments with NE errors, the next step was to classify them according to the MQM typology. In that sense, it is important to mention the information about the historical dataset used to train the NEEP model, described in the Methodology section. As stated, the model was trained based on the following error types: “Address Format”, “Currency Format”, “Date/Time Format”, “Measurement Format”, “Number Format”, “Telephone Format” and “Wrong Named Entity”.

When analysing the test dataset, used to evaluate NEEP, we found different error types, as can be seen in **Table 8**. The table consists of NE errors in the dataset and the number of errors the model did not identify and the ones it identified. They are classified by error type, according to the MQM framework.

**Table 8 - NE Error types: NEEP v1**

| | **NEEP v1 captured the NE error?** | | |
|---|---|---|---|
| **Which error type** | No | Yes | Grand Total |
| Address format | 12 / 0.52% | | 12 / 0.52% |
| Currency format | 57 / 2.46% | 4 / 0.17% | 61 / 2.64% |
| Date/time format | **90 / 3.89%** | 55 / 2.38% | **145 / 6.27%** |
| Measurement format | 89 / 3.85% | 6 / 0.26% | 95 / 4.11% |
| MT hallucination | | 15 / 0.65% | 15 / 0.65% |
| Number format | 31 / 1.34% | 39 / 1.69% | 70 / 3.03% |
| Wrong NE | **1,186 / 51.28%** | 729 / 31.52% | **1,915 / 82.79%** |

| | | | |
|---|---|---|---|
| **Grand Total** | **1465** | **848** | **2313** |

Source: NEEP v1 test dataset.

Among the error types in **Table 8**, some of them present interesting features about the model performance. Taking "Address format" as the first example, we can see that, in total, there were only twelve segments containing NE errors and, from those segments, the NEEP model was not able to identify any of them. **Example 1**[23] shows that the machine hallucinated the address name when translating it and the model did not catch it as an error. In addition, this example also illustrated quite well a failure from the NER system in identifying a NE and anonymizing it at ADDRESS-0, so that the MT would have hallucinated.

**Example 1** EN: [**montanha**] azul, 8020
ES-LATAM: [**Archivo**] Azul, 8020 (MT hallucination - address)

On the other hand, in the category "MT hallucination", the model performed really well and, from fifteen segments, it identified all of them, as in **Example 2**.

**Example 2** EN: Hello [**<<<zhanshen123456**],
ZN-CN: 您好, [ ] (MT hallucination - NE missing)

Besides **Examples 1** and **2**, two other error types highlight in the dataset analysis: "Date/time format" and "Wrong NE". Those error types have the biggest number of segments; in other words, we have more data classified as "Date/time format" and "Wrong NE" to test the model than the rest of the error type categories.

Although the NEEP model was able to identify 2.38% out of a total of 6.27% of "Date/time format" errors, it failed in most of the cases, and did not identify the NE errors in 3.89% of them. In **Example 3** we can see a MT error in the localization of the date, from English to Portuguese. It is also relevant to mention that in the source text there were two dates and in the target not only the machine did not localise, but it also omitted parts of the date. The NEEP model was not able to identify any of the two date errors in the target text.

[23] For this example, we used a random fake address, in order to be GDPR compliant.

**Example 3** EN: It was only valid between [**21.01.2022**] and [**24.01.2022**].

PT: Foi válido apenas entre as letras [**2022**] e [**24.01.2022**]. (Date/time format - Wrong date localization)

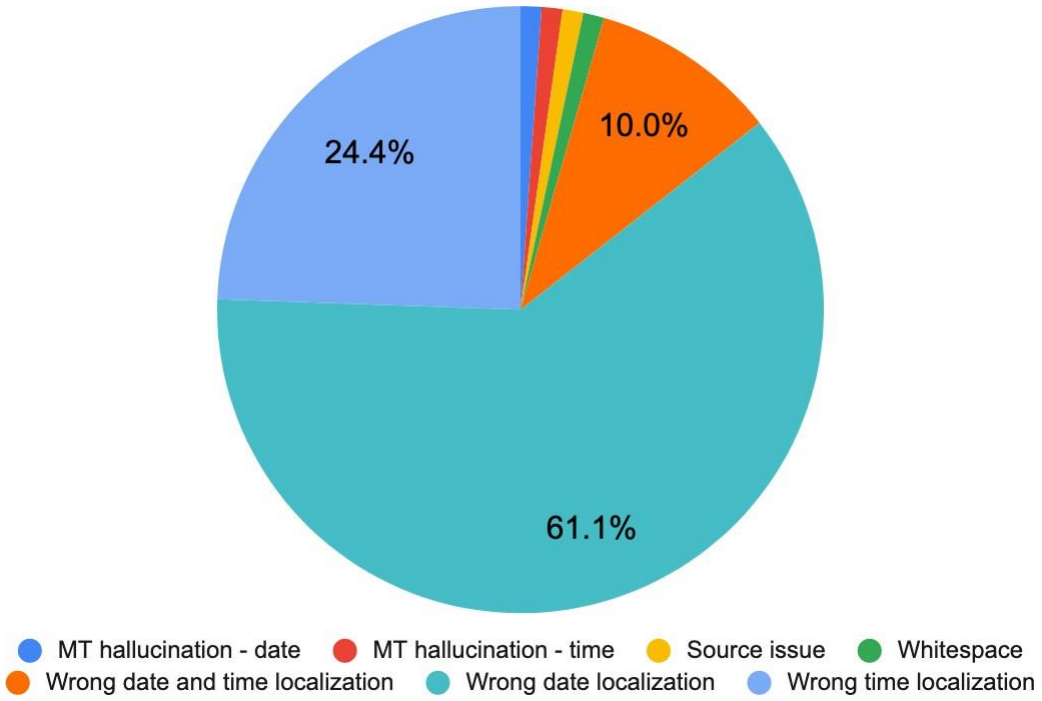
Following the same tendency as “Date/time format”, the error type classified as “Wrong NE” also presents a good result in the model performance, with 31.52% out of the total of 82.79% of the errors in NE being identified by NEEP v1. Nevertheless, in 51.28% of the cases the model failed in catching the errors, which means it still needs improvements. In **Example 4**, although we have a date, we classify it as “Wrong NE” because the machine failed and missed the year in the target text. The NEEP model should have identified it, but it did not.

**Example 4** EN: As per the tracking link, it was delivered on [**December 20, 2021**].

FR: Selon le lien de suivi, il a été livré le [**20 décembre**]. (Wrong NE - year missing)

Taking the categories “Date/time format” and “Wrong NE” as the ones presenting the higher percentage of segments in the entire dataset, 6.27% and 82.79% respectively, we chose them to illustrate the breakdown per error type within each category.

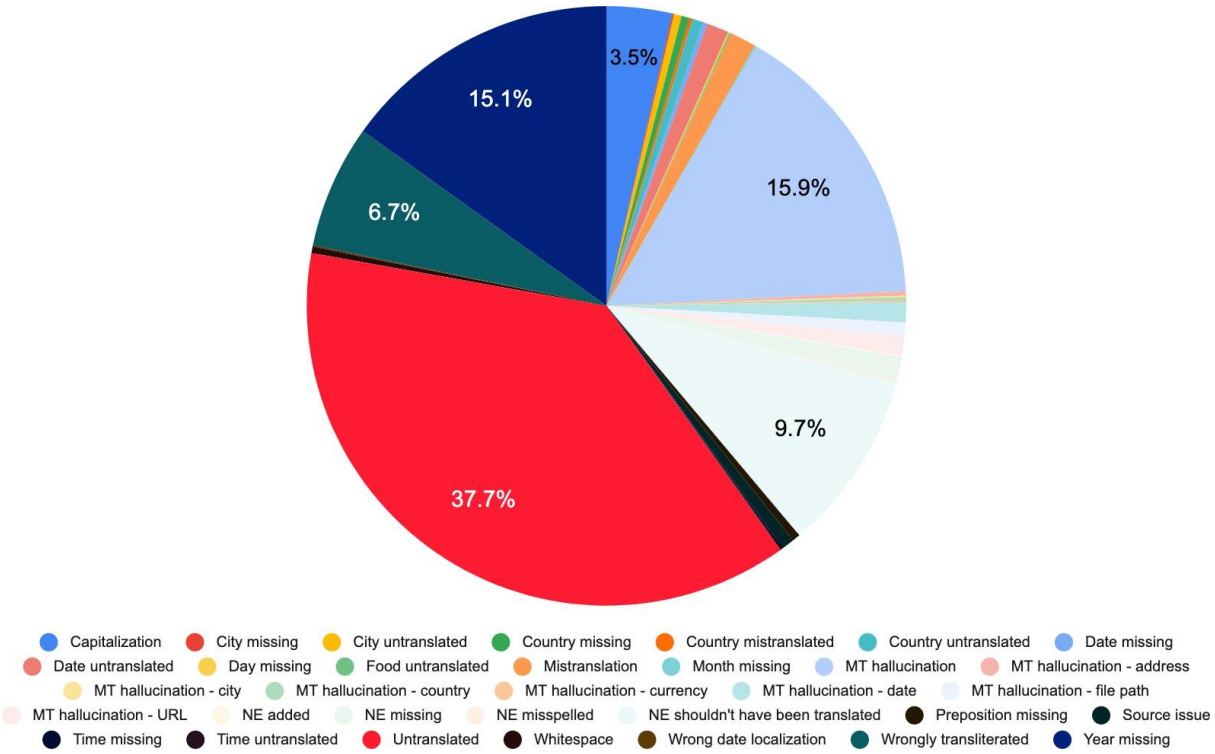
**Figure 14 - Date/time Format Errors**




Source: NEEP v1 test dataset.

**Figure 14** represents all the types of errors that occurred in the "Date/time format" MQM category in the dataset. Among all of them, we can highlight the date and time as the named entities being more inclined to have localization errors - date with 61.1% and time with 24.4% - and also the ones NEEP fails the most in identifying. This analysis was made considering only the date and time in algorithms. Which means that when there was an error in the date or the time in written form, they were classified as "Wrong NE". Moreover, it gives us a wider perspective about how MT deals with numbers in general and, consequently, NEEP performance in identifying these errors.

**Figure 15 - Wrong NE Errors**




Source: NEEP v1 test dataset.

In **Figure 15**, we have all the error types attributed to "Wrong NE". Although the amount of errors is considerably higher than in **Figure 14**, representing "Date/time format", we must say that, in the dataset, we have a higher number of segments classified as "Wrong NE" (see Table 4). The most common error within "Wrong NE" is related to "untranslated" named entities, counting the percentage of 37.7% of errors; and is followed by "MT hallucination", corresponding to 15.9% of the errors. In addition, there are many variations of MT hallucination we found through the analysis of the segments.

When we cross the information of **Figure 14** and **Figure 15**, there is one type of error that strikes out. It is interesting to notice that the model is not able to properly identify errors in dates, whether it is related to numbers, or when it comes to the written form. In that sense, apart from the MT hallucinations in other related named entities, we can also find that 1.10% out of all the "Wrong NE" errors are still related to dates.

In the next section we are going to focus on the errors distributed by languages in order to verify what are the most common errors for each language, what are the languages more inclined to have MT errors and the NEEP v1 model performance in identifying them.

### 5.3. BREAKDOWN PER LANGUAGE

Not only were we able to evaluate the NEEP model performance by error types, as described in the previous section, but also we could evaluate and track what languages were more prone to suffer certain types of errors in the MT and, among those, what languages NEEP failed the most in identifying the errors.

In order to have an overview of the languages, before zooming in on the specific error types per language, we display in **Table 9** the 6 languages with more segments in the entire dataset. When comparing the total number of segments per language from the dataset with those containing NE errors - before they go through the NEEP model evaluation - we have the numbers that are considerably high in the column “Number of segments with NE errors”. After running the NEEP model in the dataset, we have the following results for those languages:

**Table 9 - Number of segments per target languages**

| Target language | Total number of segments | Number of segments with NE errors | Number of NE errors NEEP captured |
|---|---|---|---|
| KO | 403 | 391 (97.02%) | 161 (41.18%) |
| FR | 270 | 219 (81.11%) | 27 (12.33%) |
| AR | 268 | 257 (95.89%) | 100 (38.91%) |
| ZH-CN | 177 | 171 (96.61%) | 72 (42.11%) |
| RU | 151 | 137 (90.72%) | 75 (54.74%) |
| DE | 151 | 128 (84.76%) | 21 (16.41%) |

Source: NEEP v1 test dataset.

When analysing the information in **Table 9**, it becomes evident that there is a consistent presence of certain language pairs, both in the total number of segments and in the number of segments containing NE errors. On that matter, it is important to add that there is a disparity in the amount of segments per language, as we have shown in **Figure 6** (**section 4.1**), and that it may influence the percentage of errors in each of them. We can even see in **Table 10** the languages with a smaller number of segments in the dataset.

**Table 10 - Languages with a small number of segments**

| Target language | Total number of segments | Number of segments with NE errors | Number of NE errors NEEP captured |
|---|---|---|---|
| ID | 4 | 3 | 1 (33.33%) |
| DA | 8 | 8 | 0 |
| ZH-TW | 11 | 11 | 1 (9.09%) |
| VI | 13 | 11 | 6 (54.55%) |

Source: NEEP v1 test dataset.

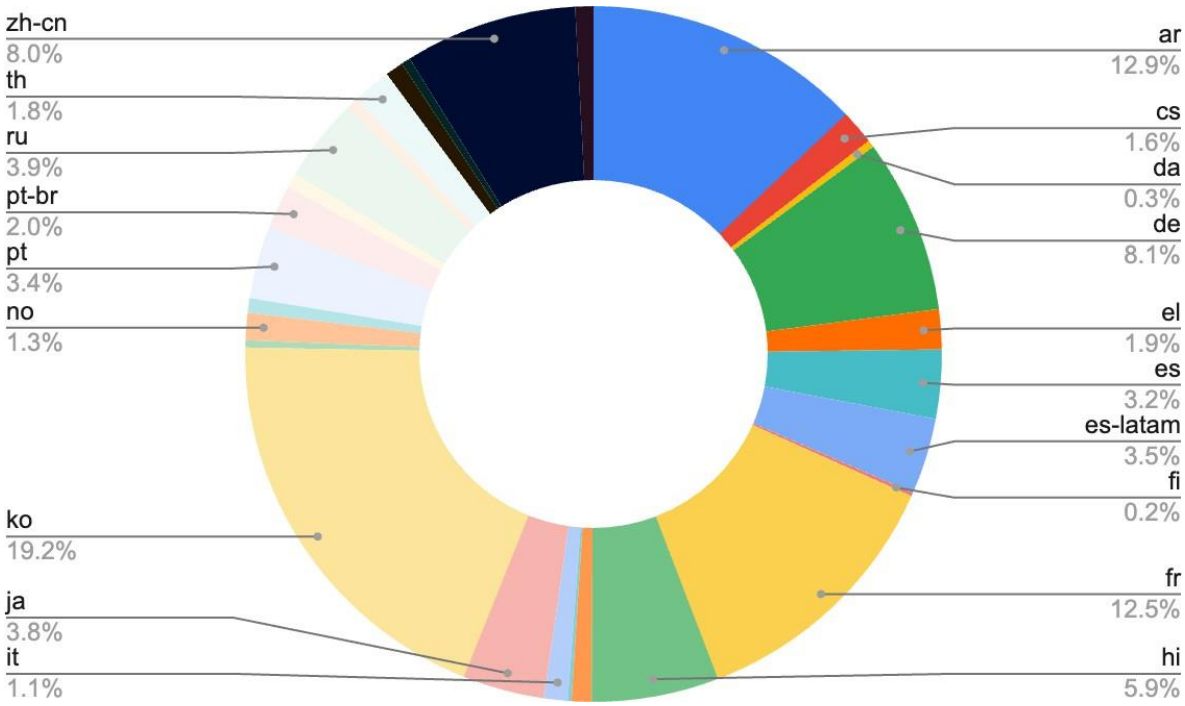
After the NEEP v1 model evaluation, and a manual validation of the errors carried out by two expert annotators, we considered relevant to detail some of the error types that were more recurrent in certain target languages. In the next subsections we are going to detail the error types per language, mainly considering “Wrong NE”, “Date/time format” and “Currency”.

### 5.3.1. WRONG NE

In **Figure 16** we can see the distribution of “Wrong NE” errors by the target languages. Korean is the language presenting the higher number of “Wrong NE” errors in the entire dataset, with over 200 “Wrong NE” errors. Nevertheless, Arabic and French also present a high number

of errors - 153 and 148, respectively. Not surprisingly, they are also the languages presenting a high number of segments in the dataset. In addition, within the "Wrong NE" category, there are three main common errors: 1) NE are untranslated, 2) NE hallucinated, and 3) dates omission[24] (mainly prepositions and/or years).

**Figure 16 - Percentage of Wrong NE errors per Language**




Source: NEEP v1 test dataset.

Despite the classification by error types, there is another aspect that must be mentioned when we analyse the target languages. The languages more inclined to MT errors are the same ones that require transliteration, because they have a different alphabet other than the Latin script. Therefore, Arabic, Chinese, Korean and Russian appear as the ones presenting more errors.

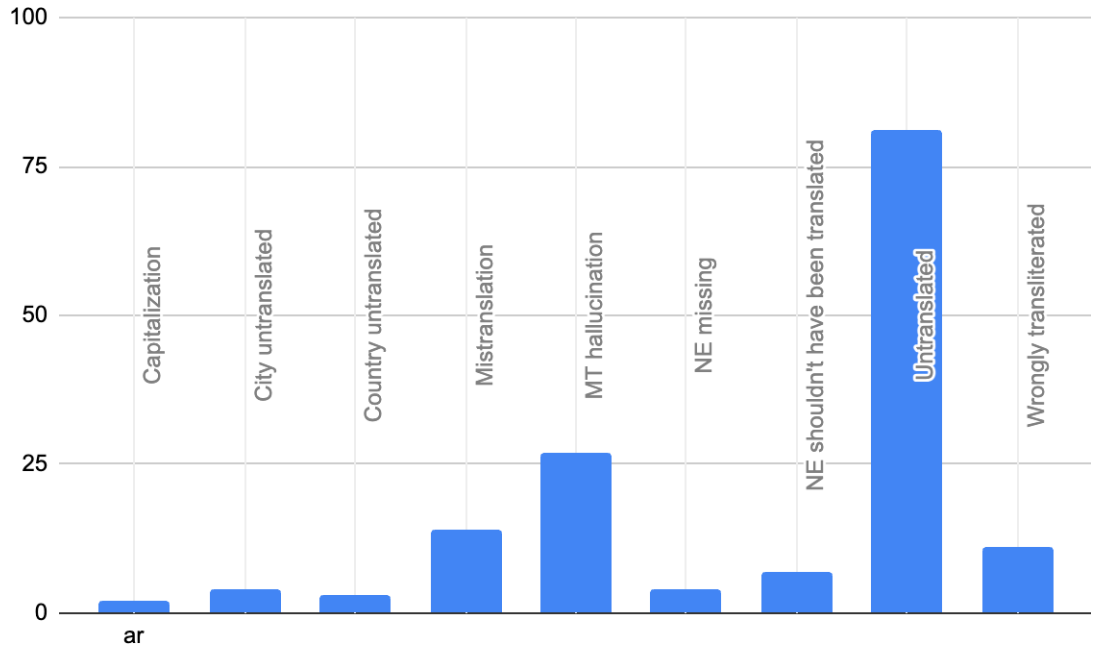
Tackling Arabic, we can see that from a total of 268 segments, 95.89% (see **Table 9**) were classified as having NE errors. The NEEP model had a high performance in 100% of the

[24] When it comes to dates, we classify them as "Wrong NE" when they are in their full written form; and we classify them as "date/time format" when they are written in the numerical form.

segments not containing NE errors, because, indeed, it did not identify them as having errors. On the other hand, when we turn our attention to the number of segments classified as having NE errors, the NEEP model only identified 38,91% of them - as we show in **Table 9** - and failed in 61,09% of the segments, not identifying them as having NE errors.

The highest percentage of errors in Arabic was in "Wrong NE" and, in **Figure 17**, we can see fine-grained information about the types of errors we were able to find in that category throughout this language. Undoubtedly, "untranslated" highlights among the other error types, because Arabic is a language that demands transliteration. Furthermore, not only names were not translated, but there are also some cases of untranslated city and country names.

**Figure 17 - Wrong NE errors in Arabic**




Source: NEEP v1 test dataset.

The second more common error type in Arabic was "MT hallucination". "MT hallucination" happens when the NE was translated as another word other than the one in the source language; or, even, when the MT does not translate the NE and changes it into another word in the target language. In **example 5** we can see the MT made two errors: 1) not transliterating the name "Andrade", and 2) hallucinating in the parts "Juliana de" of the name.

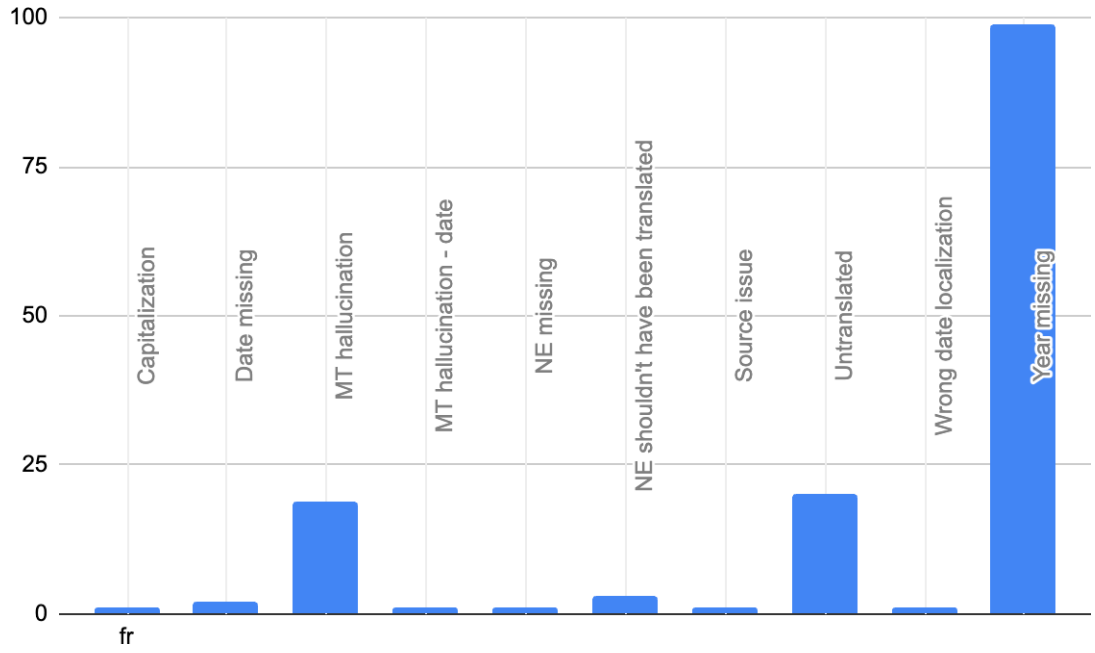
**Example 5:** EN: Juliana de Andrade[25]

AR: يوليا تفعلا Andrade (Wrong NE - MT hallucination)

[yulia du 'andrade]

Differently from Arabic, which uses a non-Latin script, French uses the Latin script and it reflects in the difference of occurrence of "Wrong NE" errors, as can be seen in **Figure 18**. Although there is a certain amount of errors classified as "MT hallucination" and "Unstranslated", it is discrepant the number of "Years missing" – 99 out of 148 errors in total.

**Figure 18 - Wrong NE errors in French**




Source: NEEP v1 test dataset.

**Example 6:** EN: I have checked and I have found that your return has reached us on December 23, [**2021**].

FR: J'ai vérifié et j'ai constaté que votre retour nous est arrivé le 23 Décembre [ ]. (Wrong NE - Year missing)

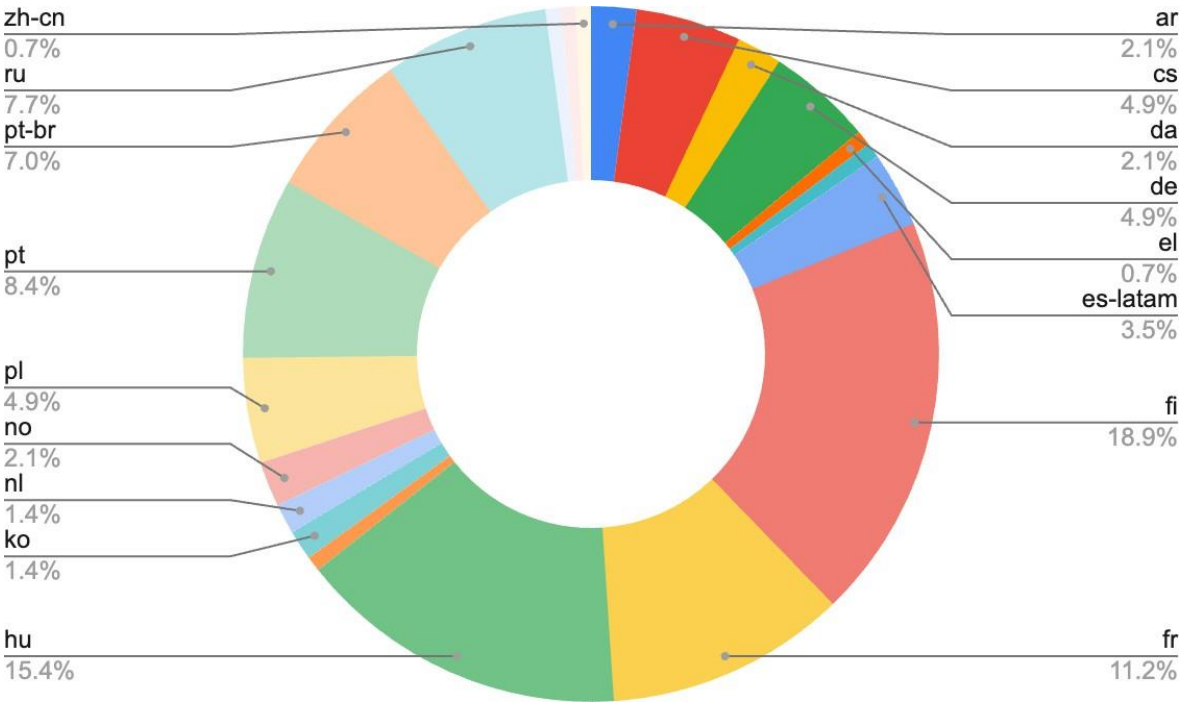
[25] For this example, we used a fictional name, in order to be GDPR compliant.

### 5.3.2. DATE/TIME FORMAT

Contrarily from what we detailed about the “Wrong NE” errors, the interesting fact about “Date/time format” is that the languages with more errors in this category are not the ones with more segments in the dataset: Finnish - 43 segments in total, and 88.37% of them containing NE errors -; and Hungarian - 46 segments and all of them containing NE errors.

For “Date/time format”, we mainly considered errors related to numbers and how those numbers should be localised in the target language. Thus, we must take into account that each language has to follow its own specificities when it comes to localisation. In **Figure 19** we can see the distribution of errors in “Date/time format” by target language. Although the languages from different scripts are represented in the figure, such as Korean and Russian, it is surprisingly interesting to see that the highest number of errors in “Date/time format” occur in Latin script languages, such as Finnish, French, Hungarian and Portugueses, for instance.

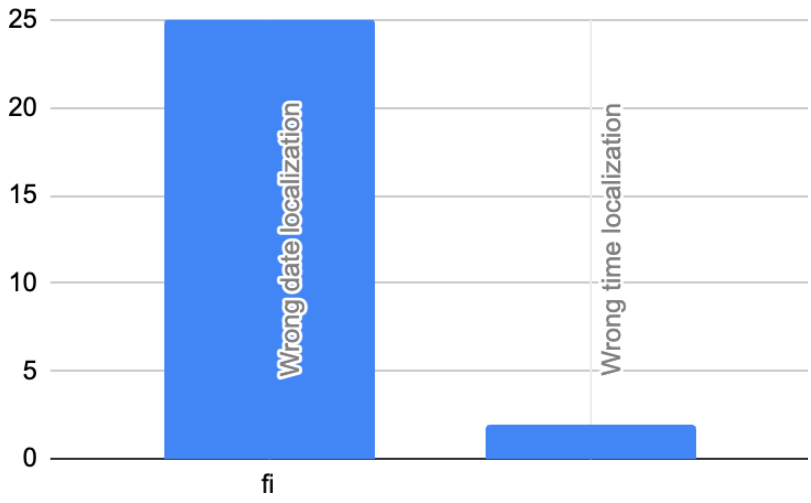
**Figure 19 - Number of Date/time format errors per Language**




Source: NEEP v1 test dataset.

Most of the errors in this category happen when there are numbers. In the analysis of the dataset we noticed that at least one of the numbers will hallucinate or have a wrong localization. In this case we are not properly talking about dates and hours written in the full form.

**Figure 20 - Date/time format errors in Finnish**



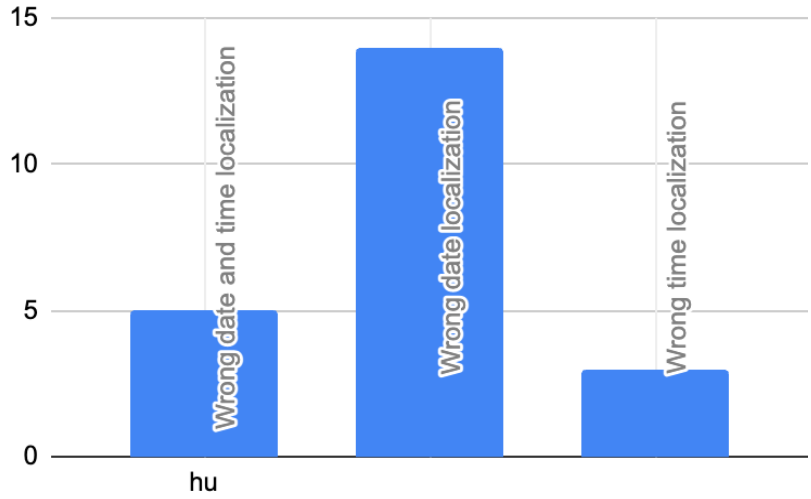

Source: NEEP v1 test dataset.

In **Figure 20** we have the "Date/time format" errors in Finnish. All of the errors were in the localisation. According to Unbabel's guidelines for localization, "Date/time format" in Finnish should follow the model "d.m.yyyy". In other words, when the source text goes to the MT step, the expectation is to get the localization of the date in the corresponding target language format. The error in **Example 7** refers not only to the date not being localised but also NEEP failing in identifying the present error.

**Example 7:** EN: I can see that you have redeemed the gift card [**19-06-2021**], so you can expect to receive it within the next two days.
FI: Näen, että olet lunastanut lahjakortin [**19-06-2021**], joten voit odottaa saavasi sen seuraavien kahden päivän kuluessa. (Date/time format - Wrong date localization)

More interestingly, when we analyse the case of Hungarian, as in Finnish, we have a higher amount of errors in date localisation and a fewer number in time localisation. However, we also have a certain amount of errors in date and time localisation when both of them appear in a segment, as we can see in **Figure 21** and in **Example 8**.

**Figure 21 - Date/time format errors in Hungarian**




Source: NEEP v1 test dataset.

Example 8 illustrates quite well what we previously stated about the errors in date and time localisation in the same segment. After having analysed Unbabel's guidelines for localization of "date/time format" in Hungarian, we see it should follow the model "yyyy. mm. dd.", for dates and "hh.mm" or "hh:mm" for time.

**Example 8:** EN: I have checked the tracking for your order once more and I can see, that the parcel was delivered to you address yesterday [**(09.02.2022)**] at around [**17:55 PM**].
HU: Még egyszer ellenőriztem a megrendelés nyomkövetését, és látom, hogy a csomagot tegnap [**(09.02.2022)**] az Ön címére kézbesítettük [**17:55 PM**] körül. (Date/time format - Wrong date and time localization)

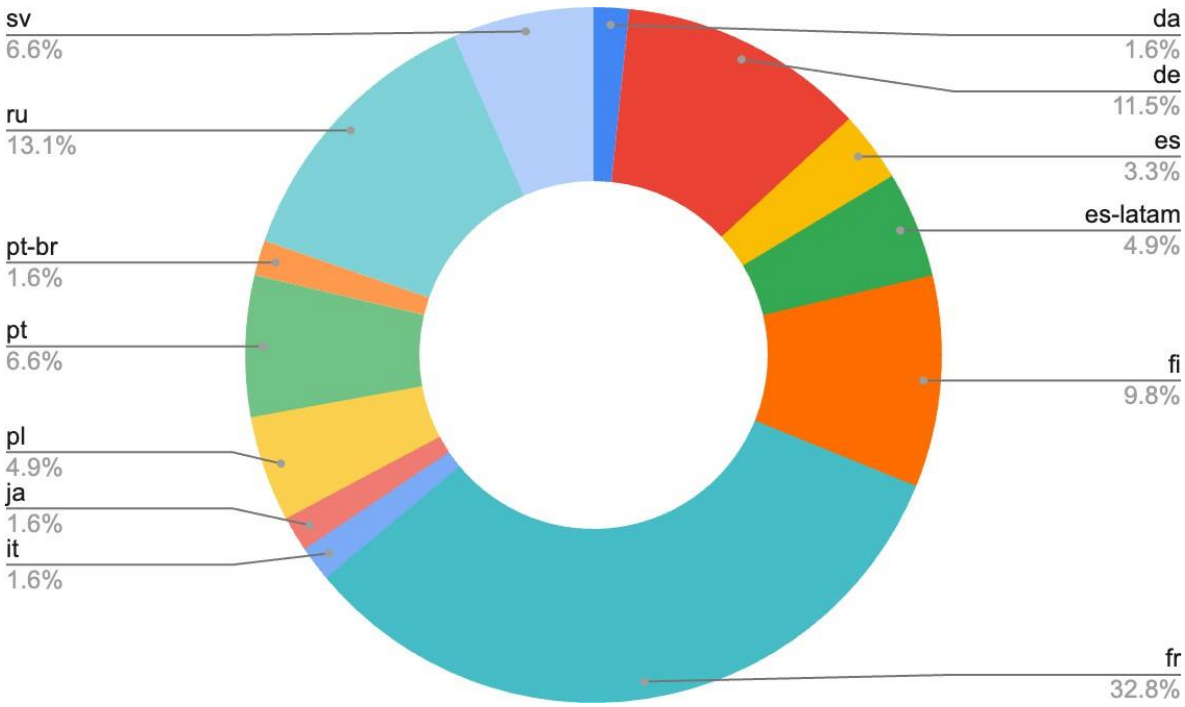
Nevertheless, as we see in **Example 8**, neither the date or the time were localised. This is especially remarkable when we analyse those named entities written in Roman numerals; which means that when they are in the full written format, they decrease significantly.

### 5.3.3. CURRENCY

Since we were describing the "Date/time format" errors, focusing on the numerical issues, it is interesting to analyse some errors in the "Currency format" as well. Those errors mainly occurred in the localization of the currency in the target language. As we have already described, the localization process involves writing the NE in the correct format in the target language. When we consider the currency role in a given text, its localization must be well done, since its errors may cause, among other things, loss of money for a company, for instance.

As shown in **Figure 22**, the languages presenting more errors in the currency format are French, with 32.8%, followed by Russian, counting 13.1% and German, with 11.5%.

**Figure 22 - Number of Currency format errors per Language**




Source: NEEP v1 test dataset.

Despite the fact that French is the language presenting more errors in “Currency format”, we decided to tackle Russian (**Example 9**) and German (**Example 10**) as instances of this error type, as a way to bring a variety of languages to the examples given along this work. Following Unbabel’s guidelines for Russian, the currency localization was well performed considering the currency symbol. Although Russian uses a different alphabet, they also accept “EUR” which is a ISO convention accepted worldwide.

On the other hand, one of the biggest issues, that also occurred in German (**Example 10**), involved the hallucination of the decimal separator, whether it was a comma (,) or a full stop (.). In **Example** 9, we can see that the MT did not localise the decimal separator to Russian and, moreover, the NEEP model failed in identifying this error.

**Example 9:** EN: Total Cost: [**280.40 EUR**].
RU: Общая стоимость: [**280.40 EUR**]. (Currency format - Wrong currency localization)

For German, in **Example 10**, something quite similar to Russian happened. Not only did the MT not localise the decimal separator, as it should do, but it also failed in the localisation of the currency symbol. As stated by the Unbabel’s guidelines for German, the localisation should use the currency symbol “€” after the currency value. Consequently, we can consider the NEEP model failed not identifying both issues in the German currency localization.

**Example 10:** EN: One of the returned items of this order which is LOCK UP TEE - Printed T-shirt - vanilla, we have already refunded the amount of [**5.60 Eur**] to you on January 27, 2022.
DE: Einer der zurückgesendeten Artikel dieser Bestellung ist LOCK UP TEE - Printed T-Shirt - vanilla, wir haben Ihnen bereits am 27. Januar den Betrag der [**5.60 Eur**] zurückerstattet. (Currency format - Wrong currency localization)

## 5.4. DATA AUGMENTATION: OVERALL RESULTS

Results for the NEEP v2 performance demonstrated that the new version of the model not only performed quite well, but also it overcame the NEEP v1 model results. Out of the total of 3,164 segments in the sample we chose to analyse, 1,806 of them had a certain type of NE error, and the new model was able to capture 1,261 of these segments, as displayed in **Table 11**. It represents a good amount of 69.82% of the segments from the entire sample dataset.

Nevertheless, when there is not any type of NE error in the segments, the NEEP v2 performance tends to slightly decrease, when compared to the NEEP v1. From a total of 1,358 segments presenting no NE errors, the model caught 15.32% of them. Yet, it does not represent an overall bad performance, since 84.68% not containing NE errors were not annotated by the model v2. This leads us to a substantial improvement in the recall results, as displayed in **Table 12**.

**Table 11 - Amount of NE errors: NEEP v2**

| | **NEEP v2 captured the NE error?** | | |
|---|---|---|---|
| **Is there a NE error?** | No | Yes | Grand Total |
| No | 1,150 / 84.68% | 208 / 15.32% | 1,358 |
| Yes | 545 / 30.18% | 1,261 / 69.82% | 1,806 |
| **Grand Total** | **1,695 / 53.57%** | **1,469 / 46.43%** | **3,164** |

Source: NEEP v2 test dataset.

Differently from the NEEP model v1, model v2 significantly improved its performance in the recall results, achieving 0.6982. Though precision decreased a little bit, compared to model v1 (**Table 7**), it continues to represent a good amount, reaching 0.8584. Therefore, not only the new model (v2) tends to perform better at identifying segments containing NE errors, but also NEEP v2 succeeds in correctly identifying them. A way to measure it is by analysing how much the F-measure increased from 0.5360 in the NEEP v1 to 0.7701 in the NEEP v2. It is remarkable that the NEEP v2 model performs better than its first version.

**Table 12 - Performance measurement: NEEP v2**

| TP | TN | FP | FN | Precision | Recall | F-measure |
|---|---|---|---|---|---|---|
| 39.85% | 36.35% | 6.57% | 17.23% | 0.8584 | 0.6982 | 0.7701 |

Source: NEEP v2 test dataset.

The following analysis aims at comparing both models' results in the identification of NE errors. Thus, we are going to focus on the number of NE errors both models captured in the sample dataset we used to carry out the last testing phase. Additionally, we are going to highlight the most remarkable improvements of the NEEP v2 model.

### 5.5. THE NEEP MODEL COMPARISON: V1 *versus* V2

After analysing the general performance of the NEEP v2 model, we separated in **Table 13** the comparison analysis of the two versions of the NEEP model, so as to understand their strengths and weaknesses, and establish how much the NEEP v2 model improved its performance in capturing NE errors.

**Table 13** presents the total number of segments we used to test NEEP v2 and is divided according to the question: "NEEP captured the NE error?", and the performance of NEEP v1 and NEEP v2. The compared results show a clear and considerable improvement of the NEEP v2 model.

**Table 13 - Comparison of the NEEP models**

| NEEP captured the NE error? | NEEP V1 | NEEP V2 |
|---|---|---|
| No | 2,487 / 78.60% | 1,695 / 53.57% |
| Yes | 677 / 21.40% | 1,469 / 46.43% |
| **Grand Total** | **3,164** | **3,164** |

Source: NEEP v2 test dataset.

The NEEP v1 managed to capture 21.40% of the NE errors, out of the 3,164 segments, on the other hand, the NEEP v2 went much further and managed to capture 46.43%, over the double

percentage of v1. Nevertheless, if we take the amount of NE errors not captured by the models, we must consider that both models still need improvement. Though, even in that case, the NEEP v2 continued to outperform the NEEP v1. Out of the 3,164 segments, NEEP v1 failed in identifying 78.60% of them, as the NEEP v2 only failed in 53.57% of the cases.

### 5.6. THE NEEP v2 IMPROVEMENTS PER NE ERROR CATEGORY

Given the remarkable improvement of the NEEP v2 model that we described in the previous section (**section 5.5**), we must now analyse the results obtained by error types and the distribution of those errors by language. **Table 14** presents all the error categories found in the testing dataset and indicates how many errors of each type were correctly identified ("Yes") or not identified ("No") by the model. In addition, we can see in **Table 14** the performance of the NEEP v2 model in identifying NE errors in different categories, according to the MQM framework.

**Table 14 - NE Error types: NEEP v2**

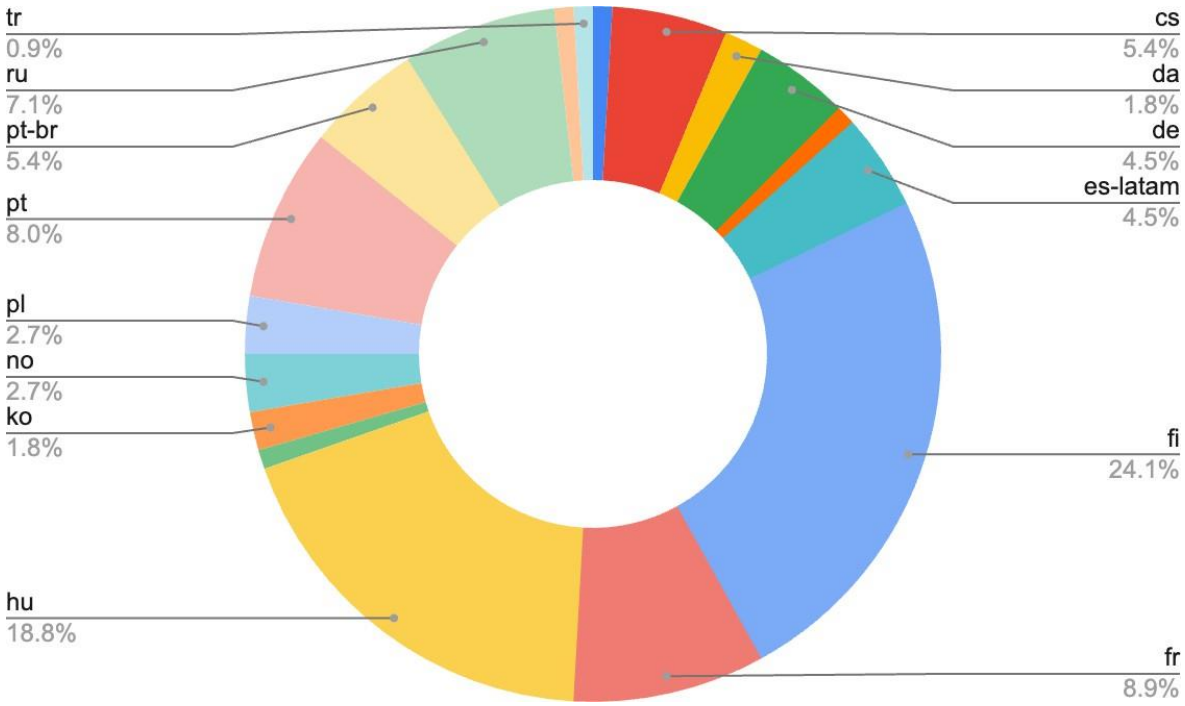
| | **NEEP v2 captured the NE error?** | | |
|---|---|---|---|
| **Which error type** | No | Yes | Grand Total |
| Address format | **3 / 0.55%** | **9 / 0.71%** | **12 / 0.66%** |
| Currency format | 1 / 0.18% | 32 / 2.54% | 33 / 1.83% |
| Date/time format | **3 / 0.55%** | **112 / 8.88%** | **115 / 6.37%** |
| Measurement Format | 66 / 12.11% | 29 / 2.30% | 95 / 5.26% |
| MT hallucination | | 15 / 1.19% | 15 / 0.83% |
| Number format | 10 / 1.83% | 44 / 3.49% | 54 / 2.99% |
| Wrong NE | **462 / 84.77%** | **1,020 / 80.89%** | **1,482 / 82.06%** |
| **Grand Total** | **545 / 30.18%** | **1,261 / 69.82%** | **1,806** |

Source: NEEP v2 test dataset.

Similarly to the results we analysed from the first test of the NEEP v1 model in **section**

**5.2** (see **Table 8**), the test carried out with the NEEP v2 also presents a very significant performance identifying NE errors in the categories "Date/time format" and "Wrong NE". Notwithstanding, NEEP v2 goes beyond the NEEP v1 and presents a good performance also identifying "Address format" errors (see **Table 14**), which have not been identified by the first version of the model.

**Figure 23 - Number of Date/time format errors per Language**



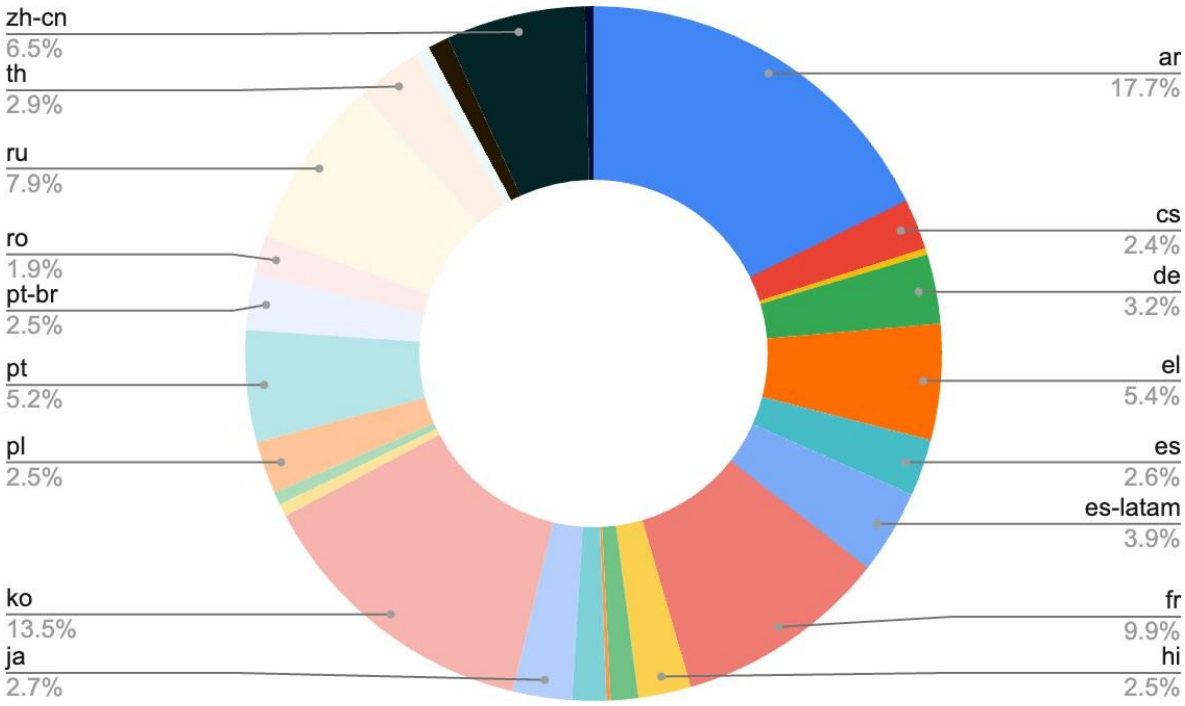

Source: NEEP v2 test dataset.

Taking "Date/time format" as the first category to analyse, we must start from the 115 segments containing that type of NE errors. That number represents 6.37% of the segments in the test dataset identified as containing NE errors. Out of the 115 segments, the model was able to properly identify 8.88% of them, having failed only in 0.55% of the segments containing "Date/time format" errors.

**Figure 23** shows the 8.88% of "Date/time format" errors divided by language. We have two most prominent languages: Finnish, with 24.1% of NE errors identified as "Date/time format"; and Hungarian, with 18.8% of the NE errors in the same category. Compared to **Figure**

**19**, we have a slightly significant improvement in the NEEP v2 performance at identifying this error type.

**Figure 24 - Number of Wrong NE errors per Language**



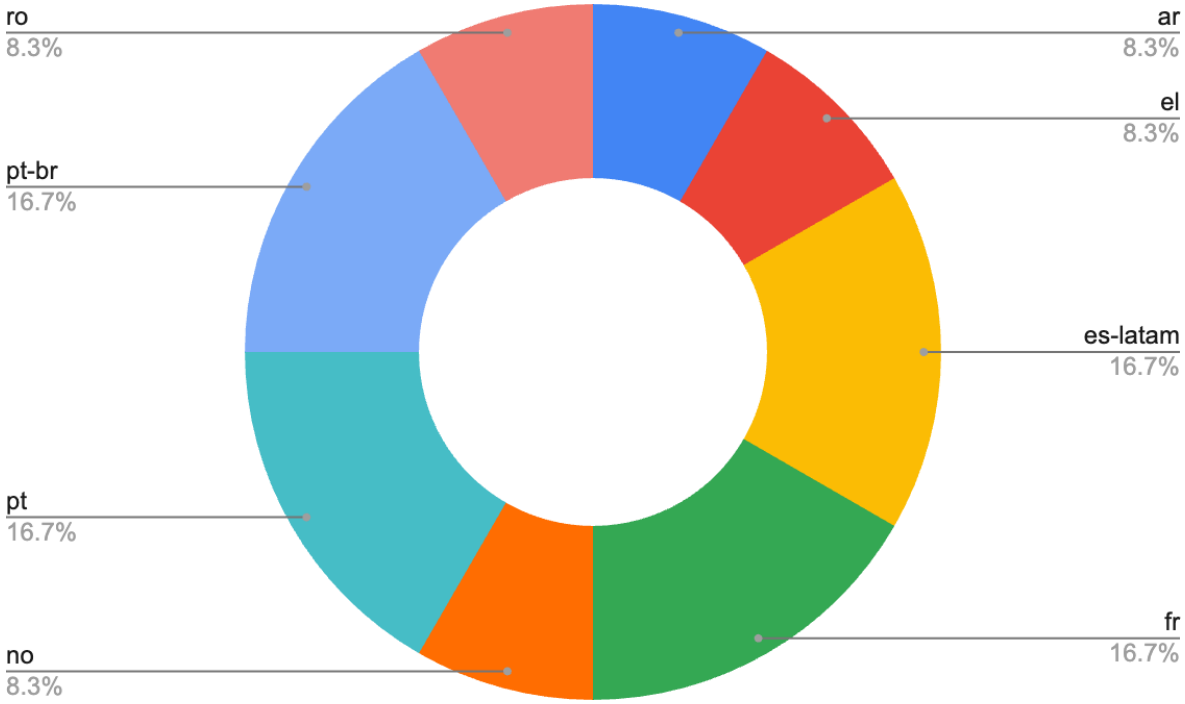

Source: NEEP v2 test dataset.

The category containing a wider number of segments with NE errors is certainly “Wrong NE”. This error category not only stands out in the number of segments in the dataset but also in the number of language pairs presenting that type of error. In Table 14, out of the 1482 segments identified as containing “Wrong NE” errors, 80.89% of them were well captured by the NEEP v2 model.

More interestingly, the new version of the model presents such good performance that the languages identified as having more “Wrong NE” errors were the ones which needed special treatment because they have a script different from Latin. Arabic and Korean are the languages presenting more identified NE errors in the “Wrong NE” category: the first with 17.7% of the errors, and the former with 13.5% of the errors. If we compare the results presented in **Table 9** and **Figure 16** to the ones we obtained after the data augmentation and the retesting of the

model, we can see an obvious improvement of the NEEP v2 for the languages more prone to NE error.

**Figure 25 - Number of Address format errors per Language**




Source: NEEP v2 test dataset.

Finally, differently from the results presented in **section 5.2** (see **Table 8**), we have some significant results for the category "Address format". In the first version and testing of the NEEP model, it was not able to identify any of the segments containing NE errors for "Address format". After the data augmentation, the NEEP v2 model gave an output of 0.71% of the "Address format" identified, out of an amount of 12 segments containing NE errors (**Table 14**). Though it still needs refinements and more segments to improve its NE errors identification, it already represents a significant gain for the present research.

Another important information to highlight is the number of languages the NEEP v2 model was able to identify "Address format" NE errors in. **Figure 25** shows a variety of eight languages containing "Address format" NE errors that the model succeeded in identifying. Among them, pt-br, pt, es-latam and fr present better results. For future work, we believe that we

can retrain the model with more data on “Address format”, so as to improve its abilities in different language pairs.

### 5.7. REMARKABLE DIFFERENCES BETWEEN THE MODELS

After analysing the overall performance of the two versions of the NEEP model - separately and compared - we considered fundamental to make a fine-grained analysis of where exactly were the differences between them. Tackling the error type differences, we built **Table 15**, in order to show what were the cases where the NEEP v2 achieved better performance over the NEEP v1. Therefore, **Table 15** presents a breakdown of different error types according to the MQM framework we used to validate the errors both NEEP models were able to identify.

As we described in **section 5.5**, the NEEP v2 outperformed the first version of the model. Thus, to understand the strengths and weaknesses of the new model, we analysed, from the 44 segments that presented more than one NE error type per sentence, what were the ones where NEEP v2 improved regarding v. On one side of the table, we have the error types annotated in the NEEP v1 analysis. Note that for each segment classified with only one error type by the NEEP v1, we have a corresponding classification carried out in the NEEP v2 output. That means the NEEP v2 went further and was able to identify more NE errors in segments the NEEP v1 only managed to identify one NE error.

The majority of segments, 77.27%, are annotated as containing “Wrong NE” errors, identified by the NEEP v1. Nevertheless, when we breakdown the analysis into error types the NEEP v2 was able to identify, for the same number of segments, the model managed not only to identify more errors, but also different error types within each segment. More interestingly, in **Table 15** we can see that five segments in which the NEEP v1 only identified “Wrong NE” errors, the NEEP v2, resounding to what we showed in **section 5.6** (see **Table 14** and **Figure 24**), also identified “Address format” errors, counting 11.36%.

**Table 15 - Breakdown per error type**

| Which error type - NEEP v1 | Which error type - NEEP v2 | Number of segments |
|---|---|---|
| Currency format | Wrong NE \| Date/time format \| Currency format | 1 / 2.27% |
| Currency format Total | | 1 / 2.27% |
| Date/time format | Date/time format \| Currency format | 1 / 2.27% |
| | Date/time format \| Number format | 1 / 2.27% |
| | Wrong NE \| Date/time format | 5 / 11.36% |
| Date/time format Total | | 7 / 15.91% |
| MT hallucination | Date/time format \| MT hallucination | 1 / 2.27% |
| MT hallucination Total | | 1 / 2.27% |
| Number format | Date/time format \| Number format | 1 / 2.27% |
| Number format Total | | 1 / 2.27% |
| Wrong NE | Date/time format \| Number format | 1 / 2.27% |
| | Wrong NE \| Address format | 5 / 11.36% |
| | Wrong NE \| Currency format | 3 / 6.82% |
| | Wrong NE \| Date/time format | 21 / 47.73% |
| | Wrong NE \| Date/time format \| Number format | 1 / 2.27% |
| | Wrong NE \| Number format | 3 / 6.82% |
| Wrong NE Total | | 34 / 77.27% |
| **Grand Total** | | **44** |

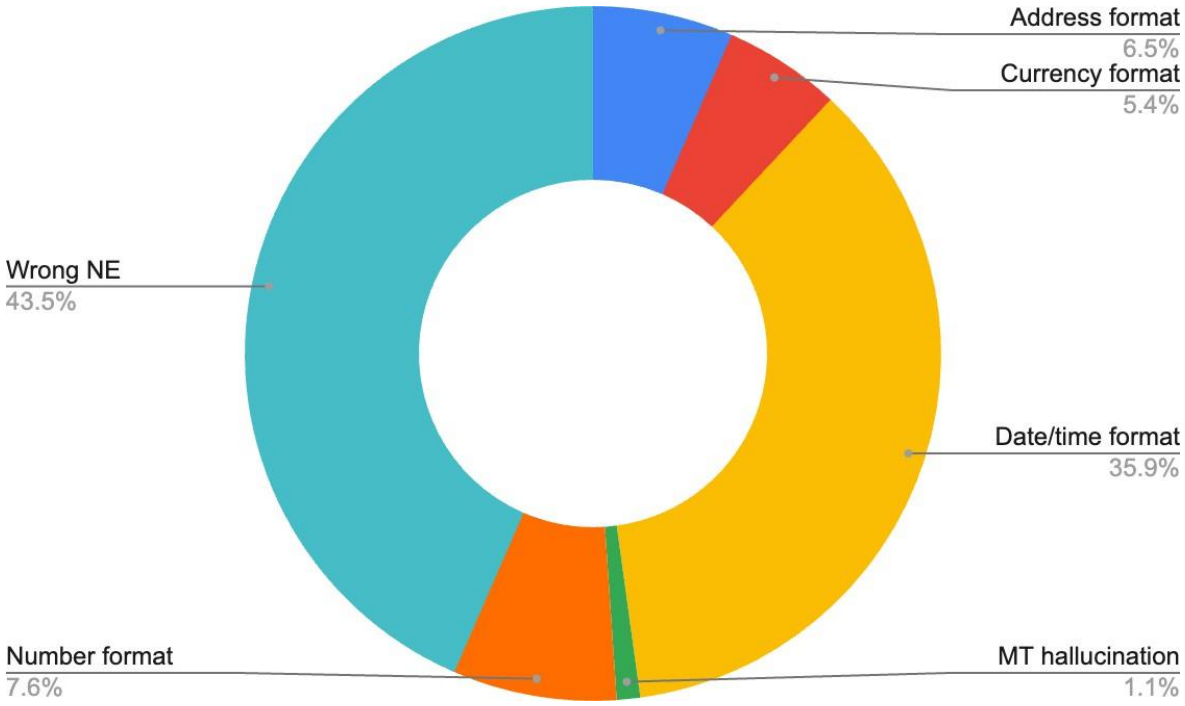
Source: NEEP v2 test dataset.

Another aspect to consider is the amount of “Date/time format” error the new model improved in identifying. Across **Table 15**, we can see that in almost all of the 44 segments we analysed that the NEEP v2 identified more than the NEEP v1 in each segment, there are “Date/time format” errors. It is interesting to note that the new model has a good sensibility to identify issues with the localisation of numbers, especially when it comes to “Date/time format”.

Besides, what is more interesting is the NEEP v2 model’s ability to identify different types of NE errors in the same segment. If we take the case of segments that the NEEP v1 model identified as only presenting “Wrong NE” errors, and compare to the NEEP v2 performance, in

21 segments, the NEEP v1 only identified “Wrong NE” errors. On the other hand, for the same 21 segments, the NEEP v2 went further and not only identified the “Wrong NE” present in the segments, but also “Date/time format” errors which represent the amount of 47.73% of the segments misidentified by the first version of the model, and improved in the second version, considering the 44 segments in **Table 15**.

**Figure 26 - Number of error types outperformed by the NEEP v2**



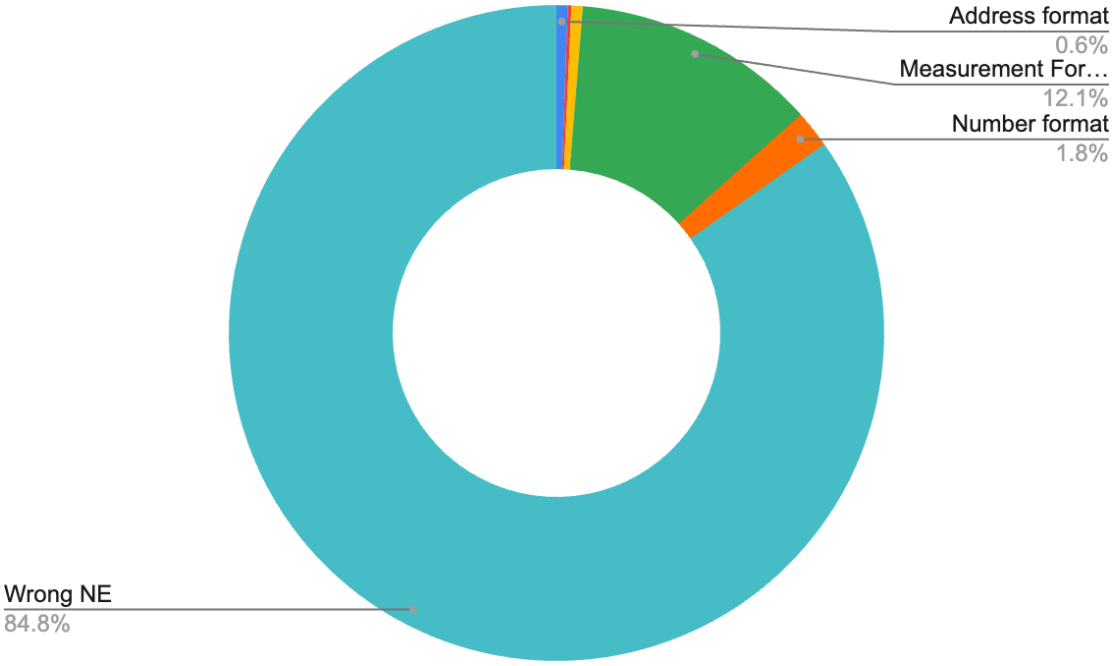

Source: NEEP v2 test dataset.

From a total of 69.82% of NE errors that the NEEP v2 succeeded in capturing out of the entire dataset, 44 segments presented more than one NE error type, as we have just described (see **Table 15**). In **Figure 26**, we made a breakdown of those segments by error type, in order to better understand what they represent in terms of volume of errors. “Wrong NE” (43.5%) and “Date/time format” (35.9%) are certainly the categories presenting more errors identified. Nevertheless, we must reinforce remarkable improvements of the NEEP v2, identifying errors in the categories “Address format” and, also, “Currency format”, achieving 6.5% and 5.4% of the additional errors identified by the model respectively.

## 5.8. ASPECTS TO IMPROVE IN THE NEEP V2 MODEL

Given the overall results of the NEEP v2 model analysis (see **Table 11**), from a total of 30.18% of NE errors that the NEEP v2 failed in capturing, we have two main categories that need to be better addressed, in **Figure 27**. The first of them is "Wrong NE" and the second is "Measurement format".

**Figure 27 - Number of error types the NEEP v2 needs improvement**



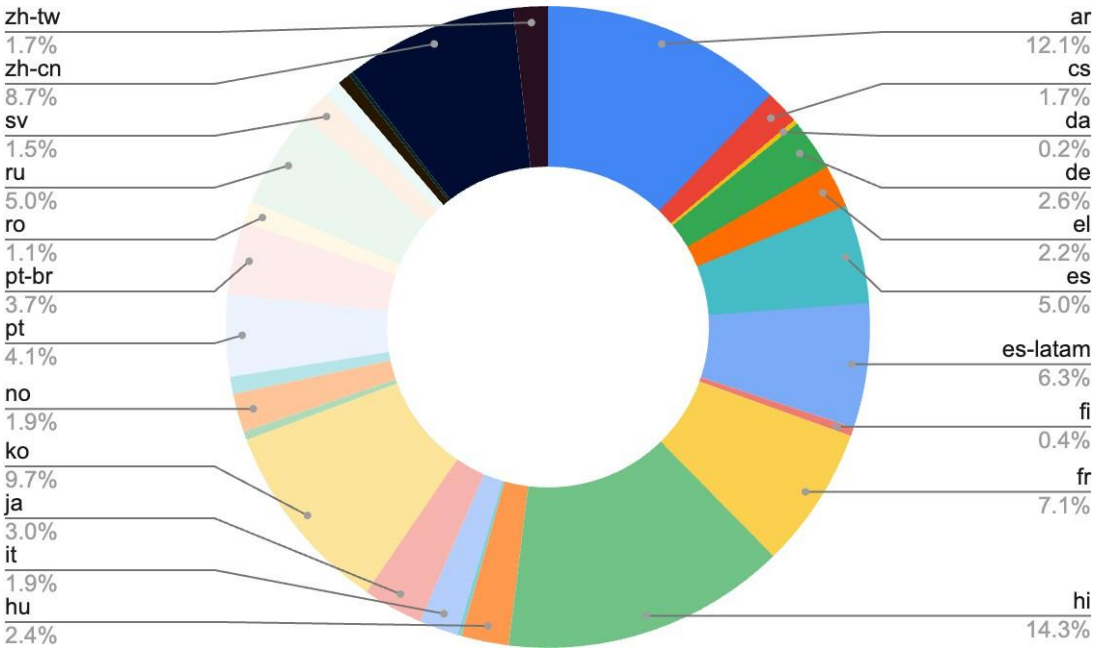

Source: NEEP v2 test dataset.

The case of "Wrong NE" errors is relevant to be pointed out, because it is certainly the category presenting more data in the dataset. That might be analysed from two perspectives: a) at the same time it is the category that the model achieves better performance at identifying NE errors; b) it is also the one more error prone, due to the amount of data and the variety of entities that can be classified as "Wrong NE"[27]. Naturally, in **Figure 26**, "Wrong NE" presents 84.8% of NE errors not identified by the model, out of the 30.18% in the total amount of data (see **Table 14**).

[27] We have this information detailed in **Chapter 3**, **section 3.3.4**.

In **Figure 28**, we can see the "Wrong NE" errors separated by language. Although we have considerable improvements (described in **section 5.6**) and those improvements have a higher representation in the total performance average of the model, we still have space for enhancing the model. Not only in languages with different script, such as hi (14.3%), ar (12.1%) and ko (9.7%), but also in Latin script languages such as fr (7.1%) and es-latam (6.3%), for example.

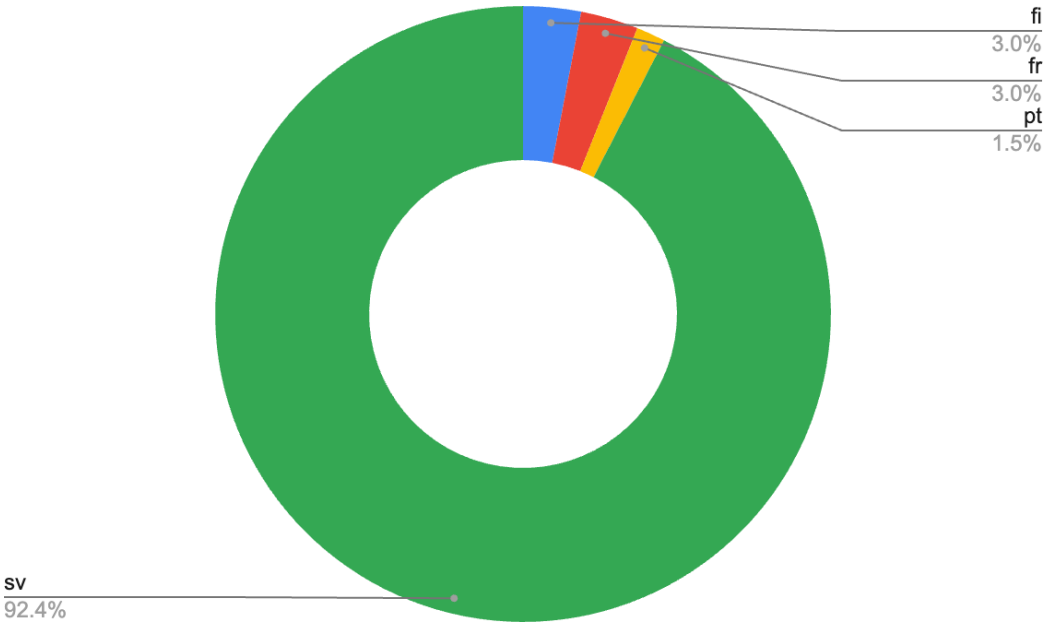
**Figure 28 - "Wrong NE" errors per language**




Source: NEEP v2 test dataset.

In addition, in **Figure 27**, we can also see a considerable percentage of not identified NE errors in the category "Measurement format" (12.1%). "Measurement format" errors are usually related to the localisation of the numbers used in the measurement standards. Though the NEEP v2 model failed in identifying NE errors in a certain number of segments for this category, we still consider the model had a good performance. When analysing **Figure 29**, we can see that model only failed in four different languages.

**Figure 29 - "Measurement format" errors per language**




Source: NEEP v2 test dataset.

Among the languages in **Figure 29**, sv represents 92.4% of the NE errors in "Measurement format". According to Unbabel's Language Guideline, Sweden uses the metric system. Therefore, we should maintain the original format of measures from the source text and avoid conversion.

**Example 11:** EN: Consume low power and offer an ultra-small footprint of [**1.6mm x 0.8mm**] dimensions with 0.55mm pitch.
SV: Har låg strömförbrukning och extremt litet avtryck på [**1,6 x 0,8 mm**] med 0.55mm stiftavstånd. (Measurement format - Wrong measurement localization and whitespace)

**Example 11** illustrates quite well the issue involving the localisation of NE. Although the MT performs a good translation, maintaining information accurately, when it comes to the numbers indicating measurement, we can see clear errors in the use of commas, measurement unit missing and unnecessary whitespace. The NEEP v2 model failed, by not identifying those

errors. However, from this data augmentation and the NEEP v2 analysis we can tackle even more accurate data to improve the model's performance.

# 6. CONCLUSIONS AND FUTURE WORK

The objective of this thesis was to analyse the first version of the NEEP model, and, from there on, explore the data augmentation path, so as to improve the model's performance for several languages and domains. The process resulted in the development of a new version of the model (NEEP v2), which was also tested and proved to have great improvements (over 20 F-measure increased performance).

In the first experiment we analysed and evaluated the fine-grained performance of the NEEP v1 model. We took into account all the issues with the language pairs - especially the differences between Latin script languages and other types of script, such as Russian and Asian languages, for instance, when analysing the MT output. In addition, we made a breakdown through error types in order to identify the best approach in the data augmentation step of the thesis.

The evaluation consisted in annotating the NE errors in the segments from the testing dataset, according to the MQM framework. In total, for this first experiment, we analysed 4177 segments, from different clients and domains. We found that the NEEP v1 model failed the most in the categories "Wrong NE", "Date/time format" and "Currency", the two last categories being more related to number issues and localisation in the target language. Moreover, we also noticed a higher recurrence of errors in languages such as Arabic, for instance.

Thus, for the data augmentation step, which was subsequent to the first analysis of the NEEP v1 model, we created a dataset with all the available NE errors in several languages, mainly focusing on the error types that the NEEP v1 model failed the most in identifying. We gathered annotated data from the NEETS dataset, which included both MT and PE. The task concerned introducing errors related to NE into the dataset. This error induction was aimed at increasing the NEEP model's sensitivity to NE errors, especially in the categories where it previously performed poorly.

To induce these errors, we used a script called *smaug*. In practice, *smaug* automatically replaces the original NE in the source input with alternative NE, resulting in a set of perturbed data that we used for retraining the NEEP model. In total, we collected an amount of 2,112 segments, in different languages and domains, thus creating a new dataset with generated NE perturbations. Moreover, we also eliminated sentences from the original training dataset that did not contain NE errors. This adjustment resulted in an increase in the proportion of sentences with

NE errors from 0.1% to 5%. With the data augmentation and the retraining of the NEEP model we were able to train a new version of the model, NEEP v2, which we then tested and evaluated.

The last step of the thesis was to assess and evaluate the performance of the NEEP v2 model. We selected a sample from the NEEP test dataset comprising 3164 segments. This enabled us to compare the NEEP v2 model's outputs with those generated by the NEEP v1 model. We achieved great results by increasing the recall to 0.6982 and the precision, although not having increased, still maintained a high value of 0.8584. An overall improvement can be measured by examining the increase in the F-measure score from 0.5360 in NEEP v1 to 0.7701 in NEEP v2. It is noteworthy that the NEEP v2 model demonstrates superior performance compared to its initial version, since it increased over 20 points in the F-measure results.

The NEEP v2 model identified 69.82% of the NE errors in the testing dataset, failing in 30.18% of them. Among those NE errors where the model achieved a good performance, we have the categories "Wrong NE", "Date/time format" and "Address format". "Wrong NE" and "Date/time format" were categories we aimed at improving with the data augmentation step, and it is a consistent gain to have been able to enhance them. Furthermore, the category "Address format" has achieved better results than in the first version of the model.

Developing a model that is able to predict NE errors is very challenging because of the different domains, content types and languages used at the company. For a first attempt at developing a model that was able to identify NE errors in MT outputs, we have obtained successful results. With fine-grained analysis of both v1 and v2 NEEP models and the data augmentation process, we were able to tackle important NE error issues across languages and domains. Notwithstanding, we recognize improvement is still necessary, in order to obtain a NEEP model version able to provide more consistent results for all the NE categories present at Unbabel's scope of work.

Additionally, we must consider the cases where the NEEP v2 failed as a way to maintain continuous improvement of the model and of the research on NE. Finally, we expect that in the future this NE error prediction module can be integrated into translation workflows for quality evaluation, and also in the core of the Center for Responsible AI[28] project, in order to enable better results with PII.

[28] Available at: <https://centerforresponsible.ai>.

# 7. BIBLIOGRAPHY


Agarwal, O., et. al. (2021). Interpretability Analysis for Named Entity Recognition to Understand System Predictions and How They Can Improve. *Computational Linguistics* 2021; 47 (1): 117–140. doi: https://doi.org/10.1162/coli_a_00397

ALPAC 1966 *Language and Machines: Computers in Translation and Linguistics.* A report by the Automatic Language Processing Advisory Committee. National Academy of Sciences, Washington, D.C.

Banerjee, S., & Lavie, A. (2005). METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 65–72. https://aclanthology.org/W05-0909.pdf

Castilho, S., et. al. (2018). Attaining the Unattainable? Reassessing Claims of Human Parity in Neural Machine Translation. arXiv:1808.10432v1 [cs.CL] 30 Aug 2018.

Castilho, S., et. al. (2017). Is Neural Machine Translation the New State of the Art?. The Prague Bulletin of Mathematical Linguistics No. 108, pp. 109–120.

Castilho, S., et. al. (2017). A comparative quality evaluation of PBSMT and NMT using professional translators. In *Proceedings of Machine Translation Summit XVI: Research Track*, pages 116–131, Nagoya Japan.

Chen, L., et. at. (2023). How Is ChatGPT's Behavior Changing over Time? arXiv:2307.09009v1 [cs.CL] 18 Jul 2023.

Chinchor, N., & Robinson, P. (1997, September). MUC-7 named entity task definition. In *Proceedings of the 7th Conference on Message Understanding* (Vol. 29, pp. 1-21).

Data Protection Act, 2018. *Data Protection Act 2018*. [online] GOV.UK. Available at: <https://www.gov.uk/government/collections/data-protection-act-2018>.

Gao, Y., et. al. (2023). How to Design Translation Prompts for ChatGPT: An Empirical Study. Available at: arXiv:2304.02182 [cs.CL].

Gonçalves, M. (2021). Analysis on the impact of the source text quality: Building a data-driven typology. Relatório de Estágio do Mestrado em Tradução, Faculdade de Letras da Universidade de Lisboa.

Graça, J. (2018). Unbabel: How to combine AI with the crowd to scale professional-quality translation. In *Proceedings of the AMTA 2018 Workshop on Translation Quality Estimation and Automatic Post-Editing*, pages 41–85, Boston, MA. Association for Machine Translation in the Americas. Available at: <https://aclanthology.org/W18-2103.pdf>.

Hutchins, W. J. (1995). Machine translation: A brief history. In *Concise history of the language sciences.* Pergamon.431-445.

Hutchins W. John. Machine Translation over fifty years. In: Histoire Épistémologie Langage, tome 23, fascicule 1, 2001. Le traitement automatique des langues. pp. 7-31.

Jurafsky, D., and Martin, J. H. (2014). Speech and language processing. Vol. 3. US: Prentice Hall.

Kaddour, J. et. at. (2023). Challenges and Applications of Large Language Models. Available at: <arXiv:2307.10169> **[cs.CL]**.

Kenny, D. (2018). Machine Translation, The Routledge Handbook of Translation Studies and Linguistics, J. Piers Rawling & Philip Wilson eds. London: Routledge, pp. 428-445.

Kepler, F., Trénous, J., Treviso, M., Vera, M., & Martins, A. F.T. (2019). OpenKiwi: An Open Source Framework for Quality Estimation. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 117-122. 10.18653/v1/P19-3020

Koehn, P. (2020). *Neural Machine Translation* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781108608480

Koehn, P. (2017). Statistical Machine Translation. Draft of Chapter 13: Neural Machine Translation. *Statistical Machine Translation. arXiv*.

Lommel, A., Popović, M., & Burchardt, A. (2014, May). Assessing Inter-Annotator Agreement for Translation Error Annotation. *Conference: LREC Workshop on Automatic and Manual Metrics for Operational Translation Evaluation*.

Lommel, A., Uszkoreit, H., & Burchardt, A. (2014). Multidimensional quality metrics (MQM): A framework for declaring and describing translation quality metrics. *Revista Tradumàtica: tecnologies de la traducció, (12)*. 455-463.

Lopez, A. (2008). Statistical machine translation. ACM Computing Surveys (CSUR), 40(3), 1-49.

Makhoul, J., Kubala, F., Schwartz, R., and Weischedel, R. (1999). Performance measures for information extraction. In *Proc. of the DARPA Broadcast News Workshop, Herndon, VA.*

Menezes, M. (2021). Named Entities Recognition for Machine Translation: A Case Study on the Importance of Named Entities for Customer Support. Relatório de Estágio do Mestrado em Tradução, Faculdade de Letras da Universidade de Lisboa.

Moniz, H., Torrón, M. (2020) *The Unbabel ecosystem: communities and AI-driven technologies.* Unpublished internal company document.

Mota, P., et. al. (2022). Fast-Paced Improvements to Named Entity Handling for Neural Machine Translation. In Proceedings of EAMT.

Mota, P., et. al. (2022). A Case Study on the Importance of Named Entities in a Machine Translation Pipeline for Customer Support Content. In Proceedings of EAMT.

Moorkens, J., Castilho, S., Gaspari, F., & Doherty, S. (Eds.). (2018). *Translation Quality Assessment: From Principles to Practice* (Machine Translation: Technologies and Applications ed., Vol. 1). Springer. https://doi.org/10.1007/978- 3- 319- 91241- 7

*MQM Core Typology*. (n.d.). Available at: <https://themqm.info/typology/>.

*MQM Definition*. (n.d.). QT21. Available at: <https://www.qt21.eu/mqm-definition/definition-2015-12-30.html>.

*MQM (Multidimensional Quality Metrics)*. (n.d.). Available at: <https://themqm.org/>.

NER Annotation Guidelines. (2020). Internal company document.

Nouvel, Damien, Maud Ehrmann and Sophie Rosset. 2016. Named entities for computational linguistics. ISTE.

Oliveira, D. (2010). Extraction and Classification of Named Entities. Relatório de Estágio do Mestrado em Engenharia e Ciência da Computação. Instituto Superior Técnico da Universidade Técnica de Lisboa.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, 311. https://doi.org/10.3115/1073083.1073135

Paulo, M. (2022). Analysis of context-aware Translation Memories: Part-of-Speech pattern distribution and gender neutral Translation Memories. Relatório de Estágio do Mestrado em Linguística, Faculdade de Letras da Universidade de Lisboa.

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020). Stanza: A python natural language processing toolkit for many human languages. arXiv preprint arXiv:2003.07082.

Sang, E. F., De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Silva, B. (2022). Translation Error Annotation: Building an Annotation Module for East Asian Languages. Relatório de Estágio do Mestrado em Tradução, Faculdade de Letras da Universidade de Lisboa.

Shen T, Yu L, Jin L, et al., 2022, Research on Chinese Entity Recognition Based on BERT-BILSTM-CRF Model. Journal of Qiqihar University (Natural Science Edition), 38(01): 26-32.

Steingrímsson, S. (2023). Effectively Compiling Parallel Corpora for Machine Translation in Resource-Scarce Conditions. Tese de Doutoramento em Ciência da Computação, Department of Computer Science (RU), Reykjavík University.

Rei, R., Stewart, C., Farinha, A. C., & Lavie, A. (2020). COMET: A Neural Framework for MT Evaluation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2685–2702. https://doi.org/10.18653/v1/2020.emnlp-main.213

Sirena D (2004) Mission impossible: improve quality, time and speed at the same time. Globalisation Insider 13(2.2). Available at: http://www.translationdirectory.com/article387.htm.

Snow T (2015) Establishing the viability of the multidimensional quality metrics framework. Dissertation, Brigham Young University. Available at: http://scholarsarchive.byu.edu/etd/ 5593/.

TAUS. (2015). *Harmonized DQF-MQM Error Typology*. TAUS - The Language Data Network. Available at: <https://www.taus.net/qt21-project#harmonized-error-typology>.

Wang, Z., Mayhew, S., & Roth, D. (2020). Cross-lingual ability of multilingual bert: An empirical study. arXiv preprint arXiv:1912.07840.

Zhang X, Li Y, Wang D, et al., 2019, Named Entity Recognition Based on ERNIE. Intelligent Computer and Application, 10(03): 21-26.

# 8. ANNEXES

### A. True Positives breakdown per language

| | NEEP captured the NE error? | |
|---|---|---|
| **Target language** | Yes | **% of TP per language** |
| ar | 100 | 38.91% |
| cs | 14 | 34.15% |
| de | 21 | 16.41% |
| el | 43 | 64.18% |
| es | 15 | 27.27% |
| es-latam | 36 | 41.86% |
| fi | 23 | 60.53% |
| fr | 27 | 12.33% |
| hi | 21 | 23.08% |
| hu | 14 | 30.43% |
| id | 1 | 33.33% |
| it | 12 | 42.86% |
| ja | 2 | 3.77% |
| ko | 161 | 41.18% |
| nl | 8 | 40.00% |
| no | 3 | 14.29% |
| pl | 22 | 56.41% |
| pt | 41 | 41.84% |
| pt-br | 50 | 60.98% |
| ro | 16 | 64.00% |
| ru | 75 | 54.74% |
| sv | 3 | 3.23% |
| th | 13 | 38.24% |
| vi | 6 | 54.55% |

Source: NEEP v1 test dataset.

### B. True Negatives breakdown per language

| | NEEP captured the NE error? | |
|---|---|---|
| **Target language** | No | **% of TN per language** |
| ar | 11 | 100% |
| cs | 1 | 100.00% |
| de | 23 | 100.00% |
| el | 5 | 100.00% |
| es | 7 | 87.50% |
| es-latam | 7 | 100.00% |
| fi | 4 | 80.00% |
| fr | 51 | 100.00% |
| hi | 3 | 100.00% |
| id | 1 | 100.00% |
| it | 1 | 100.00% |
| ja | 4 | 100.00% |
| ko | 12 | 100.00% |
| nl | 1 | 100.00% |
| no | 2 | 100.00% |
| pl | 2 | 100.00% |
| pt | 4 | 100.00% |
| pt-br | 10 | 100.00% |
| ru | 14 | 100.00% |
| sv | 13 | 100.00% |
| tr | 9 | 100.00% |

| | | |
|---|---|---|
| vi | 2 | 100.00% |
| zh-cn | 6 | 100.00% |
| **Grand Total** | **193** | |

Source: NEEP v1 test dataset.

### C. False Positives breakdown per language

| | NEEP captured the NE error? | |
|---|---|---|
| **Target language** | Yes | **% of FP per language** |
| es | 1 | 12.50% |
| fi | 1 | 20.00% |
| **Grand Total** | **2** | |

Source: NEEP v1 test dataset.

### D. False Negatives breakdown per language

| | NEEP captured the NE error? | |
|---|---|---|
| **Target language** | No | **% of FN per language** |
| ar | 157 | 61.09% |
| cs | 27 | 65.85% |
| da | 8 | 100.00% |
| de | 107 | 83.59% |
| el | 24 | 35.82% |
| es | 40 | 72.73% |
| es-latam | 50 | 58.14% |
| fi | 15 | 39.47% |
| fr | 192 | 87.67% |
| hi | 70 | 76.92% |
| hu | 32 | 69.57% |
| id | 2 | 66.67% |
| it | 16 | 57.14% |

| | | |
|---|---|---|
| ja | 51 | 96.23% |
| ko | 230 | 58.82% |
| nl | 12 | 60.00% |
| no | 18 | 85.71% |
| pl | 17 | 43.59% |
| pt | 57 | 58.16% |
| pt-br | 32 | 39.02% |
| ro | 9 | 36.00% |
| ru | 62 | 45.26% |
| sv | 90 | 96.77% |
| th | 21 | 61.76% |
| tr | 12 | 100.00% |
| vi | 5 | 45.45% |
| zh-cn | 99 | 57.89% |
| zh-tw | 10 | 90.91% |
| **Grand Total** | **1465** | |

Source: NEEP v1 test dataset.

### E. Augmented error types by language pairs

| | **NE error** | | | | | | |
|---|---|---|---|---|---|---|---|
| Language pairs | Date/time format | Measurement Format | MT hallucination | Number format | Untranslated | Wrong NE | Grand Total |
| en-ar | | | 1 | | | 26 | 27 |
| en-de | | | | | | 8 | 8 |
| en-el | | | | | | 17 | 17 |
| en-es | 1 | | | | | 1049 | 1050 |
| en-es-latam | | | | | | 14 | 14 |
| en-fr | | | | | | 28 | 28 |
| en-it | | | | | | 8 | 8 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| en-ja | | | | | 1 | 5 | 6 |
| en-ko | | | | | | 27 | 27 |
| en-nl | 8 | | | | | 113 | 121 |
| en-pl | | | | | | 11 | 11 |
| en-pt-br | | | | 18 | | 129 | 147 |
| en-ru | | | | | | 285 | 285 |
| en-sv | | 236 | | | | 83 | 319 |
| en-th | | | | | | 3 | 3 |
| en-vi | | | | | | 1 | 1 |
| en-zh-cn | | | | | | 35 | 35 |
| **Grand Total** | **9** | **236** | **1** | **18** | **1** | **1842** | **2107** |

Source: Augmentation dataset.

### F. Segments containing multiple error types

| Which error type - NEEP v2 | Number of segments |
|---|---|
| Date/time format \| Currency format | 1 |
| Date/time format \| MT hallucination | 1 |
| Date/time format \| Number format | 3 |
| Wrong NE \| Address format | 5 |
| Wrong NE \| Currency format | 3 |
| Wrong NE \| Date/time format | 26 |
| Wrong NE \| Date/time format \| Currency format | 1 |
| Wrong NE \| Date/time format \| Number format | 1 |
| Wrong NE \| Number format | 3 |
| **Grand Total** | **44** |

Source: NEEP v2 test dataset.

### G. Number of error types outperformed by the NEEP v2

| Which error type - NEEP v2 | Number of errors outperformed |
|---|---|
| Address format | 6 |
| Currency format | 5 |
| Date/time format | 33 |
| MT hallucination | 1 |
| Number format | 7 |
| Wrong NE | 40 |
| **Grand Total** | **92** |

Source: NEEP v2 test dataset.