Project CALL

CALL group 1

Gyum Cho Rosca Maxim Thomas Bostelaar Xianzhi Wu

***Abstract*—This report outlines the development and implementation of a language correction system for non-native English speakers. Utilizing the transformative Text-to-Text Transfer Transformer (T5) model, the system aims to identify and correct grammatical errors within written text. The methodology involves data processing, error analysis using ERRANT (ERRor ANnotation Toolkit), and the T5's text-based approach. The T5 model, particularly the T5 v1.1 - LM adapted version, is selected and fine-tuned for grammar correction tasks. The model's efficiency, and balanced performance make it an ideal candidate for addressing grammatical inconsistencies, providing a unified solution for various natural language processing (NLP) tasks. By utilizing T5, the system seeks to enhance user writing proficiency and reduce biases, contributing to a more effective and reliable language correction tool for non-native English speakers. Additionally, ethical considerations, transparency, and fairness are incorporated in the design to ensure responsible AI application.**

I. Motivation

The project started with how language complexity can make communication complicated. English is a key global language for job and business opportunities, but the way it's used varies a lot around the world. This can be frustrating, especially for people who aren't native English speakers. Our team, made up of people from different language backgrounds, experienced this with personal experience. Our diverse backgrounds showed us the small but important differences in how people construct sentences and express themselves.

We found that a big factor in these differences is the influence of a person's native language, how they learn and use English. After looking closely, we found common grammatical mistakes among non-native English speakers. For example, Japanese learners often struggle with using articles (like "a" and "the") and prepositions [1]. These mistakes aren't random, so we wanted to use this understanding to create a computer-assisted language learning (CALL) system. Our goal is to make a system that focuses on fixing these common mistakes. We know that the way someone's native language is structured affects the grammatical errors they make when learning English, so we want to build a CALL system that gives personalized feedback based on these language specific patterns.

II. Defined Goal

The primary objective of this project is to conduct an in depth analysis of texts written by non-native English speakers with the aim of identifying grammatical errors and commonly used incorrect expressions based on their mother language. This analysis involves the extraction of error frequencies, which serves as the basis for constructing a grammar correction model. The model is designed to detect recurring errors, identify typical linguistic pitfalls, and suggest improved writing practices customized to the user's language proficiency. Through the deployment of a language model and subsequent development of a web application, our team has created a comprehensive tool to provide users with insights. This tool enables users to gain a clear understanding of their writing flaws, learn from these identified mistakes, and receive guidance on how to enhance their writing skills effectively. By presenting personalized suggestions for improvement, the application empowers users to grasp the nuances of language and prevent errors in their future writing endeavors. Our goal is the creation of a Grammatical Error Correction (GEC) system specifically catering to non-native language learners. To accomplish this objective, we have divided the project into two primary sub-tasks: firstly, crafting a GEC model designed for Latin languages, and secondly, testing our methodology to build a language-specific GEC. This approach allows us to address language nuances and intricacies, offering customized solutions for diverse linguistic backgrounds and thereby empowering non-native English learners on their journey towards linguistic proficiency.

III. Method

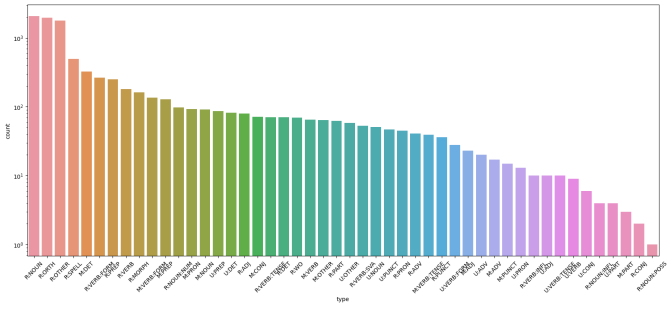

Fig. 1. Combined error frequencies for the Latin language group (log scale)

A. Data Processing

The project started by emphasizing the data processing as a foundation. The initial dataset, which had over 80,000 entries, showed some issues like duplication and non-English expressions. To address these problems, we went beyond

traditional data selection methods to ensure the accuracy of our results.

Instead of just deleting incorrect data, we focused on improving data quality. To fix duplicated data, we carefully consolidated information, making sure everything was consistent across all columns. We also gave preference to entries that provided a more comprehensive representation of the data in cases of overlap, especially regarding shared text and nationality.

We took a careful approach to removing non-English expressions, understanding that some people might use their mother languages other than English. Rather than translating these entries, we chose to exclude them to maintain the integrity of the data according to our research goals.

Additionally, we recognized the linguistic similarities among French, Spanish, and Italian, which are rooted in the Latin language system. We combined these languages into a single linguistic feature based on observed error frequencies. This approach aimed to create a more comprehensive dataset, capturing common error patterns within the Latin language group. Our goal was to enhance the precision of our analysis and contribute to a more accurate understanding of error dynamics in this linguistic cluster.

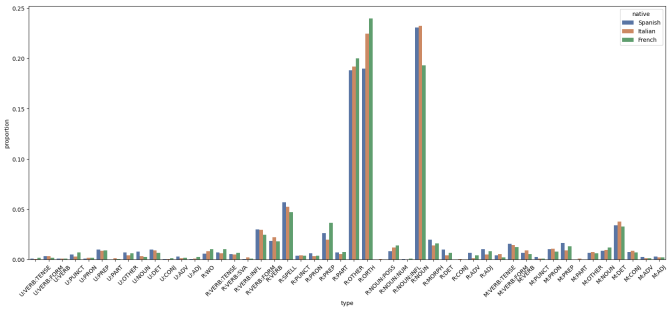

Fig. 2. Errors frequency comparison between French, Spanish and Italian

B. Expand dataset

After precisely processing the data and identifying language-specific traits, we used the T5 model to fix grammar mistakes in non-native speakers' writing. We chose Grammarly because of its high accuracy and statistical approach backed by a diverse set of examples.

Using the Grammarly model helped us correct grammatical errors in non-native speakers' text and categorize the errors based on how common and important they were. We systematically classified these observations, as shown in a visual aid, giving us a full picture of error patterns and helping us find areas where we needed more data.

Realizing that our original dataset was not big enough to train a strong language model (only about 20 percent) of the needed data), we took a strategic approach. To increase our dataset, we analyzed writing from non-native users in the Hugging Face grammar model's database. We carefully studied the error frequency patterns in the original data to ensure the new dataset had a similar distribution of error types. By aligning the error frequencies with those in the initial dataset, we successfully expanded the dataset to a sufficient size (10,000 entries), allowing us to effectively train the language model.

A really important aspect when it comes to using a predefined model to correct sentences is the risk of circular dependency. Our goal is to fine-tune another model using the right corrections. For example, we would like to feed our model with the pair original-"Hell my name is John" and corrected-"Hello, my name is John" to achieve the right results, but when using another model we can't guarantee that this correction will be made. Instead, the model might correct the original sentence to "Hell, my name is John". Feeding this pair to fine-tune our model would just make the situation worse. To solve this problem we would calculate the two highest coefficients of the most probable corrections, if the difference between them is higher than a threshold then we would use this pair [6]. Fortunately for us, the types of sentences that our dataset contained, i.e. non-English speaking people presenting themselves in English did not have many situations in which the correction was ambiguous.

C. Fine-tuning

Recognizing the challenges of creating our own model, we operate to fine-tune an existing one. After exploring options, we settled on a T5-based model from the HuggingFace database. The fine-tuning process was straightforward. Using the HappyTransformer's HappyTextToText object, we trained the model with the dataset, requiring fewer parameters than usual. The use of predefined model parameters and the constraints of the training method were simplified, and an efficient approach underscored the ease of implementing the chosen T5-based model.

D. Testing

After we developed the model, we subjected it to a testing process. There were various phases to access its effectiveness. Initially, we conducted a loss test. a standard method in machine learning assessment to measure individual prediction accuracy. Additionally, we used sentences from the TOEFL and IELTS tests, recognized in global English qualification to compare our model's outputs with those of the original model. This rigorous testing ensured a comprehensive evaluation of our model's performance, demonstrating its proficiency in addressing example problems. It also highlighted its capability to provide more formal and academically-oriented predictions compared to existing models, solidifying its credibility and efficacy.

IV. Technical depth

A. Data processing

Implementing and prove validity of data quality procedures are essential to ensure the reliability and integrity of information essential for training our language correction model. Using data quality methods such as anomaly detection, we identified irregularities within the dataset. Anomaly detection

techniques were applied to duplicate entries, and eliminate them to reduce redundancy.

This process included a systematic analysis of the data using advanced methods like regular expressions and specific character encoding detection. The goal was to create a standardized and uniform dataset, eliminating inconsistencies and build a strong foundation stages of model development and training.

We executed cleaning procedures, systematically removing non-English characters and expressions that could potentially distort the learning process of the model. The elimination of non-English sentences was done using Python libraries, namely Lang Detect and Polyglot. While Lang Detect performed better than Polyglot, both libraries were used in the dataset. An important note is the challenge encountered due to the libraries limited support for multiple sentences, requiring each text into sentences before feeding it to the language detection libraries. The effectiveness of the language detection process depends on the languages involved, and our observations revealed that an abundance of "U:OTHER" (unnecessary other words) in the Error Tagging phase signified potential areas for improvement in the cleaning process.

We obtained correct sentences from our refined dataset, a pivotal step where careful model selection played a decisive role. Multiple models, including vennify/t5-base-grammar-correction and grammarly/coedit-large, underwent rigorous testing, evaluating their quality through software that highlighted mistakes. The Vennify model had limitations in correcting "R:NOUN" (replace noun) types of errors, like "spanish language" instead of "Spanish language." Recognizing this common error made by non-English speakers, Grammarly's error correction model was the preferred choice due to its proficiency in correcting such errors.

B. Error Tagging

To generate error frequencies within the dataset, we used the ERRANT (ERRor ANnotation Toolkit). Python library to build tags and categorize errors present in the dataset. ERRANT facilitated a systematic and comprehensive error analysis by assigning errors to 25 distinct categories. It provides a structured framework for understanding and classifying types of grammatical inconsistencies. A specific rule within ERRANT was used to distinguish between three fundamental error types: Missing, Unnecessary, and Replacement errors, contributing to a better understanding of the nature and context of the identified errors.

We evaluated an example that involved the transformation of the original sentence "This are grammatical sentence." to its corrected form "This is a grammatical sentence." ERRANT's output provided predicted tags indicating the specific types of errors found within the sentence: "R:VERB:SVA, M:DET, and R:SPELL." These tags denote Replacement (R) errors related to a subject-verb agreement (VERB:SVA), Missing (M) errors associated with a determiner (DET), and Replacement (R) errors regarding spelling (SPELL). The prefixes M, U, and R, signifying Missing, Unnecessary, and Replacement errors, respectively, were further detailed by specific error codes that reflect the errors based on Part-of-Speech (POS) categories.

ERRANT's error tagging system allowed for the identification of errors but also provided a structured breakdown of error types, enabling a more detailed and precise understanding of the grammatical inconsistencies within the text data. This comprehensive error analysis facilitated by ERRANT underpinned the development of a robust error frequency assessment method crucial for enhancing the precision and accuracy of our language correction model.

C. Synthetic data generation

Having the right frequencies we can generate more data in order to have enough to fine-tune our model. We selected 10.000 mistakes with the same frequencies from a dataset called C4200M, containing pairs of correct and wrong sentences. The dataset was selected after analyzing the dataset with which the model that we want to fine-tune was trained, we made sure that these sentences were not seen by the model before. Although T5 is trained using the C4 dataset, it is different from the C4200M in the fact that it doesn't contain the corrected sentences.

We went through the first 50.000 examples from the dataset and tagged all the pairs with the corresponding errors, using ERRANT. One challenge here is that the sentences contain multiple errors, meaning that if a sentence is selected it brings multiple error tags. This observation made it difficult to select the right amount of frequencies. In the end, there were a few types of errors that weren't selected, for example, 'M:VERB:FORM', and the rest of the errors were all fulfilled, meaning that the only sentences that can be selected from some point are these errors and no others. To solve this problem we first selected the sentences that would contain the last remaining errors and we also divided them into 3 categories of importance, the most important one being 'M:VERB:FORM', as it was the least found one in the dataset that we had.

D. Model Selection

The Text-to-Text Transfer Transformer (T5), was chooses as our foundation model based on its robust, versatile speciality [2]. T5 operates on a text-to-text framework that allow a wide array of tasks to be performed using a unified model. Also, it excels in handling various natural language processing tasks due to its text-based input-output format. One key

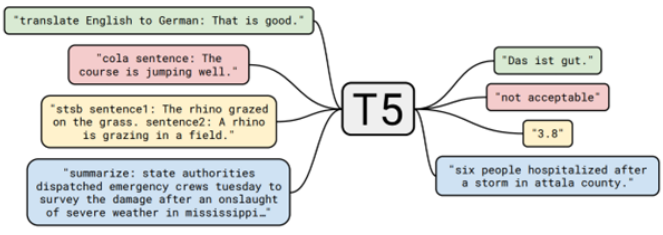

Fig. 3. Text-to-text framework

advantage of T5 lies in its text-to-text approach, represented by

a range of task specific prefixes, enabling versatile functions such as translation and summarizing. By employing a prefix-based approach, the model can perform distinct tasks through appropriate specification within the prefix. In the case of our model, the prefix 'GEC' was used to train the model explicitly for grammatical error correction.

Utilizing the Hugging Face Transformers API, we incorporated pretrained models, with HappyTransformer complementing the process for training and inference. HappyTransformer, built upon the Hugging Face transformers library, streamlines interaction with transformer models such as T5, facilitating smoother integration and utilization.

Our selection of the pretrained model involved a fine-tuned adaptation of the google/t5-small-lm-adapt model, specifically the T5 v1.1 - LM adapted version, which utilizes the T5 v1.1 small model. It is primarily trained for language modeling tasks. The T5 v1.1 - LM adapted model serves as the foundation for our grammar correction model, offering a balance between model complexity and performance. This model provides a strong foundation for addressing grammatical inconsistencies in text data. The adaptability and effectiveness coupled with its versatility and robustness in handling diverse NLP tasks influenced our decision to employ it as the framework for our grammar correction model.

The implementation of the T5 v1.1 model in our grammatical error correction system presents significant advantages compared to other existing models. One key advantage lies in its unified text-to-text approach, offering a more versatile and streamlined solution for a range of natural language processing tasks. Specifically, the T5 v1.1 model showcases substantial performance improvements in various language-oriented tasks, demonstrating competitive accuracy while maintaining a relatively smaller model size compared to larger architectures.

E. Fine-tuning

Fine-tuning an already existing T5 model is surprisingly easy. This is mostly because the architecture of the model and all its necessary parameters are already defined for you. The model that we chose, i.e. pszemraj/grammar-synthesis-small has the following important parameters: learningrate: 0.0004, gradientaccumulationsteps: 32, optimizer: Adam with betas=(0.9,0.999) and epsilon=1e-08. We changed the batch size to 8 in order to have a better ratio of memory usage and performance. The HappyTransformer HappyTextToText object allows us to further train a model using the train method. This method takes the file with the training data and the parameters as arguments. The input file format should be a CSV file that contains the input and target columns. Because of the small amount of parameters to train, only 77M, it took about 5 minutes to train the model. During the training we observed that the validation loss of the model went from 1.975 to 0.90 at the lowest point. After 2k sentences, the model validation loss started slightly to increase, going to 0.91 at the end.

In the beginning, we also tried to fine-tune another model, namely prithivida/grammarerrorcorrecterv1. This model used t5-base, meaning that it had more parameters to train than the model that we trained, for which t5-small is used. Training took about 3 hours.

Input	Output
gec: This are gramamtical sentence.	This is a grammatical sentence.

Fig. 4. Training input format

F. Web Application

In order to show that our model is deployable, as well as to provide a better experience for the possible users, we decided to build a web application. To develop our web application, we began by storing our trained model locally. Storing the model is straightforward, we used the HappyTextToText class save method after training the model. Leveraging FastAPI, we established an endpoint within the application, enabling the model to be accessed through a POST request, facilitating seamless interaction between the frontend and the model. Our process involved integrating FastAPI to create an endpoint that served as a bridge, transmitting requested data from the frontend to our stored model. This setup allowed for efficient data exchange and model utilization, ensuring a robust and responsive interface for users. By implementing this structure, our web application was adept at processing user requests and providing accurate, model-generated responses.

V. Result

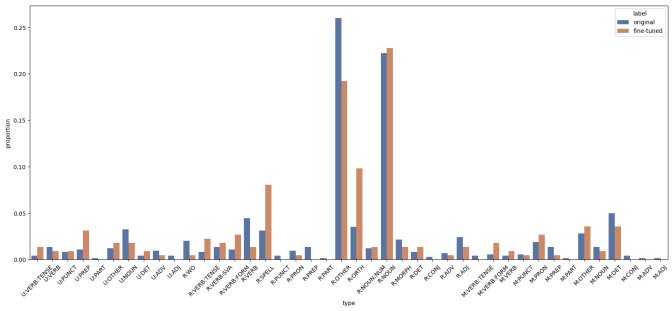

Fig. 5. Error frequency on evaluation set pszemraj/grammar-synthesis-small

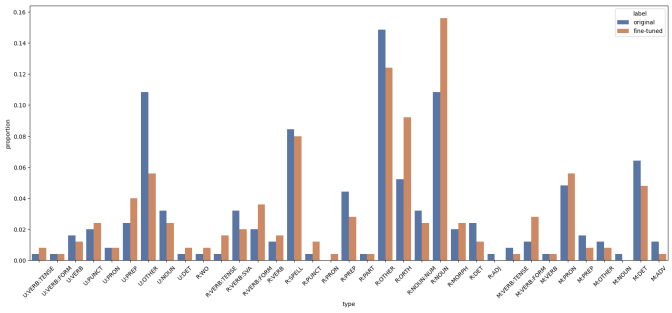

Fig. 6. Error frequency on evaluation set prithivida/grammar-error-correcter-v1

A. Result pszemraj/grammar-synthesis-small

HappyTextToText allows us to evaluate the model. We created a dataset of 10.000 mistakes and evaluated both the initial model and the old model with it and we observed the changes in frequencies from Fig 5. The dataset was selected based on the same frequencies as the training dataset from the C4200M dataset. We made sure that neither of the models had seen these sentences before.

After conducting a thorough frequency test and comparing our model with existing ones, we observed improvements in addressing major grammatical issues like "R:SPELL" (replace spelling) and "R:ORTH" (replace orthographical mistake), which can be seen in Fig 6. These types of errors are in the top 4 most frequent errors in the frequencies of non-English speakers, which can be seen in Fig. 1. These improvements indicate a positive step toward building our language correction model, in line with our goal to enhance the overall grammatical accuracy of non-native English speakers. However, it's crucial to acknowledge minor increases in other error categories.

Although we managed to increase the generation of some types of errors, we can also see that there are some frequencies that decreased, as seen in the reduction of "R:Other", even though this type of error is the third most popular among non-English speakers.

Using the eval function from HappyTextToText, we were able to see a loss of 0.968 from the original model that had 1.65.

All these metrics show a really nice improvement and promising results. By running 100 sentences through the old and fine-tuned model and computing the error types we were able to see that our model detects significantly fewer errors. In some cases, this is good, for example in the original sentence - "always for the impassible chasm of rank.", fine-tuned - "Always for the impassible chasm of rank.", original model - "Always the impassible symbol of victory.", the fine-tuned model detecting only the "R:ORTH" error type. It turns out that the old model did a lot of to change a sentence in order for it to make sense, even though it became far from the original, for example, the original sentence - "Made 1st & 20.", fine-tuned "Made 1st & 20th.", original model - "I made it last night.".

B. Result prithivida/grammar-error-correcter-v1

After conducting a thorough frequency test and comparing our model with existing ones, we observed improvements in addressing major grammatical issues like "R:NOUN" (replace noun) and "R:ORTH" (replace orthographical mistake), which can be seen in Fig 6. These types of errors are in the top 2 most frequent errors in the frequencies of non-English speakers, which can be seen in Fig. 1. Although we managed to increase the generation of some types of errors, we can also see that there are some frequencies that decreased, as seen in the reduction of "R:Other", even though this type of error is the third most popular among non-English speakers. Using the eval function from HappyTextToText, we were able to decrease in the loss as well.

All these metrics show a really nice improvement and promising results. By running 100 sentences through the old and fine-tuned model and computing the error types we were able to see that our model detects significantly fewer errors. In some cases, this is good, for example in the original sentence - "A jeweler will often adjust on your watch for a nominal cost if you prefer do not do it.", fine-tuned - "A jeweler will often adjust your watch for a nominal cost if you prefer not to do it.", original model - "A jeweler will often adjust on your watch for a nominal cost if you prefer not do it.", the fine-tuned model also detecting the "M:VERB:FORM" which is quite high in our frequencies. At the same time there are negative effects of our training, for example, original - "This is an excellent quality electric shop with unique tech offer a new comfortable close schaving feeling.", fine-tuned - "This is an excellent quality electric shop with unique tech offering a new comfortable close schaving feeling.", original model - "This is an excellent quality electric shop with unique tech that offers a new comfortable close schaving feeling.". In this example we can see that the influence of "R:VERB:FORM", the error that the model corrected, on the fine-tuned model is not positive.

VI. Conclusion

A. model evaluation

The evaluation of our language correction model involves both technical and theoretical dimensions, offering a comprehensive understanding of its effectiveness and areas for potential improvement.

From just analyzing the numbers and frequencies we can say that the methodology that we chose and the process that we did for fine-tuning the models was a success. We managed to shift most of the frequencies toward our calculated frequencies and managed to get the loss down. In order to really tell if it was a success or not we must take a look at the separate examples and actually analyze each sentence and see how it improved or not.

After analyzing the two models that we created multiple conclusions can be made. The first thing is that the choice of the model matters a lot. The first model, pszemraj/grammar-synthesis-small, was initially not meant for complex sentences that we found in the C4200M, that's why correcting them from the beginning was a struggle for this model. Because of its low amount of parameters to train, we managed to quite easily influence its weights, and shift the frequencies of the model. After training we also saw that from the 100 sentences that we tested to see the difference between the fine-tuned model and the original one there were 97 sentences that had differences. From the results, it can also be seen that in most of the cases, the sentences were better corrected than the old ones. In our opinion, this is influenced not only by the fact that the model is small, but also by the fact that the training dataset used for it, namely JFLEG. This dataset consists of less than 2k sentences. The main conclusion here is that our chosen methodology works really well on small models, partially because the small models do not work correctly from the beginning.

The second model, prithivida/grammar-error-correcter-v1, was more complex and better than the previous one. The main problem is the fact that there is a big chance that it already saw the dataset that we used, namely the C4200M, making the conclusions and results slightly inaccurate. This model, originally, had a good performance for correcting grammatical error mistakes and fine-tuning it made more sense. After training we could see that from 100 sentences that we tested the old and the new model, there were differences in only half of the sentences, the rest remaining the same. This demonstrates the fact that a more complex model is harder to influence and more data would be needed for that. Nevertheless, we could see that our fine-tuning gave some good results in most of the cases. An important observation is that a better model could be influenced badly by our frequencies, as can be seen in the results section, meaning that we have the risk of making the model worse in the cases in which the errors from our frequencies do not occur.

In conclusion, the evaluation of our language correction model highlights its technical proficiency in addressing major grammatical errors among non-native English speakers. The theoretical foundations validate its effectiveness, while identified limitations and potential improvements provide a road map for future enhancements.

B. limitation and things to improve

The first thing that needs improvement is the initial dataset that we got. Although it was good for research, in order to create a grammar error correction model we would need a dataset that would contain more complex sentences, with more complex mistakes and context. The sentences that we had were just people presenting themselves, meaning that the context of the errors was really small. This makes the frequencies inaccurate.

The process of synthetic data generation can be improved as well. The dataset that was used for training, although it had the same frequencies as the ones we initially calculated was not written by a non-English speaking person, it wasn't even written by a human, but synthetically generated, meaning that, even though the types of errors might be the same, the context in which they are is different. Another important factor is the fact that there are sentences in the dataset that even a human can't correct, meaning that in some situations we just confused the model and made the situation worse. An alternative approach, proposed for further exploration, is synthetic data generation [4]. This technique involves generating training data by introducing grammatical errors into originally correct sentences. Recent research showcases solutions capable of generating multiple errors in a single sentence [5], offering a promising avenue for expanding the training dataset efficiently. A tool that can be used is called Errgent and it offers multiple corruptions for a sentence with a given tag.

Furthermore, the current implementation utilizes the T5 v1.1 small model, the smallest variant available. Considering the availability of increased computing power, future research could explore training larger models with a greater number of parameters to potentially achieve enhanced performance. This avenue holds promise for unlocking the full capabilities of more extensive T5 model variants.

We would also pay more attention to the type of model we chose for training, just because the fact that we initially selected a wrong combination of model and dataset gave us unreliable results.

VII. Reflection about which parts of the module we have applied

A. Data processing - Data Quality

The practical application of concepts learned in the Data Quality class played a crucial role in refining and enhancing the dataset for our project. Initially, we identified various issues in the provided data, using insights from the "Detecting data quality problems" lecture. Applying knowledge gained, we systematically addressed data irregularities, including identifying and categorizing duplicated entries and non-English expressions. The complexity in distinguishing relevance, ambiguity, and dependencies of each data point, as discussed in complicating factors for detection, required subject expertise. Collaborative decision-making, especially for multi-type glitches within individual entries, involved extensive team deliberations to determine whether to correct and retain the data or discard it, aligning with data quality (DQ) principles.

Additionally, applying statistical tests based on error frequency across Italian, French, and Spanish languages revealed a systematic relationship between missing values and observed data, deviating from the Missing At Random (MAR) concept explained in class. Consequently, we systematically discarded data with missing values, aligning with insights from the Data Quality class.

In summary, the knowledge from the Data Quality class significantly influenced our approach to dataset refinement. It guided the creation of a robust dataset and equipped us with the ability to systematically categorize errors. This, in turn, boosted the accuracy of our model, emphasizing the valuable application of academic principles in real-world projects.

B. Data training

Machine learning lessons in Natural Language Processing (NLP) served as a guide and foundation for our project. The insights gained from these lessons played a key role in shaping critical decisions that influencing our model development trajectory and guiding the selection of an appropriate framework.

A paramount decision in our project was choosing the T5 model as the underlying model. Our understanding, highlighted the unique features of T5, particularly its attention mechanisms, which represent a significant advancement beyond traditional RNNs and LSTMs. This knowledge was essential in understanding how T5 effectively handles sequential dependencies by focusing on specific parts of input text during predictions. Our familiarity with the RNNs and LSTMs learned from our classes empowered us to leverage these features in constructing the foundational architecture of our project.

Furthermore, this acquired knowledge was practically applied in optimizing the T5 model. Our understanding of logistic regression models, forward passes, and back propagation played a crucial role in refining the model's performance. specifically, the training process involved sophisticated optimization through both forward passes and back propagation. Following the principles learned in our academic coursework, T5 underwent large-scale unsupervised pretraining which followed by supervised fine-tuning. The calculated gradients through back propagation were instrumental in updating model parameters, optimizing its proficiency in downstream tasks. This seamless integration of departmental knowledge facilitated informed decision making in complex areas, ensuring that challenging aspects did not impede the progress of our project.

C. Testing

Applying the foundational knowledge gained in our academic coursework, particularly in learning algorithms and loss functions, we performed testing on our T5 model. Drawing insights from our lectures, we recognized that T5, during its pretraining and fine-tuning stages, relies on learning algorithms to improve its performance. The model is strategically trained to minimize a designated loss function, like the cross-entropy loss, with the aim of enhancing its effectiveness in language-related tasks. Leveraging our understanding of learning rates and optimization algorithms, we systematically assessed the model's ability to generalize to new data. This comprehensive testing phase, rooted in the principles from our academic curriculum, allowed us to fine-tune our T5 model precisely, ensuring its proficiency and adaptability across a range of language-oriented challenges.

VIII. Ethical perspective and application of some ethical framework

The most important components on which we focused were transparency, accountability, and social impacts in our research. According to the explainable AI lectures from the university, The AI system must provide a solid and understandable explanation for its own decisions. The language correction model can be a sensitive domain since it is directly related to the user's opinions.

Ethical considerations were seriously overlooked. It is related to fairness, bias mitigation, and privacy. The AI language model should help guide the correct grammar writing for the user but should not change the original meaning of the user's opinion. One ethical challenge encountered in our model's development related to potential bias in users' language usage analysis. As the model analyzed the actual language practices of users, variations in results based on individual language patterns became different. While most outcomes remained unbiased, certain languages particularly showed a biased shape. The frequent use of emphasis words in Spanish can be an example. Emphasis words commonly occur in Spanish, such as "mucho" which means "very". The emphasis words commonly appear in text, but the AI language model trains them as strong opinions from the user. This could inadvertently influence the model's interpretation, occasionally misconstruing the intended tone of a sentence. Addressing this challenge involves a careful examination of user writing styles in different linguistic contexts to mitigate biases and foster a more universally applicable and equitable language correction system. It is particularly important in contexts concerning academic or societal matters.

Lastly, personal data from the user should be handled in an anonymous way. The personal data can include the privacy of the user. Leaking personal data can cause serious problems, such as violating a user's daily life or causing unexpected harm.

References

[1] E. Izumi, K. Uchimoto, and H. Isahara, Nihonjin 1200-nin no eigo supikingu kopasu, ALC Press, Tokyo, Japan, 2004.

[2] Bryant, Felice, and Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. https://aclanthology.org/P17-1074.pdf

[3] Raffel, Shazeer, Roberts, Lee, Narang, Matena, Zhou and Li. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer https://arxiv.org/pdf/1910.10683.pdf

[4] Stahlberg and Kumar. 2021. Synthetic Data Generation for Grammatical Error Correction with Tagged Corruption Models https://aclanthology.org/2021.bea-1.4.pdf

[5] Rahman. 2022. Judge a Sentence by Its Content to Generate Grammatical Errors https://arxiv.org/pdf/2208.09693.pdf

[6] M. Chodorow, J. R. Tetreault, and N.-R. Han, "Detection of grammatical errors involving prepositions," in Proc. ACL-SIGSEM Workshop on Prepositions, 2007.

https://github.com/GyumCho/AIProject