Identifying masculine generics in Italian

Paolo Mainardi

Alma Mater Studiorum – Università di Bologna

Dipartimento di Interpretazione e Traduzione

Forlì

26th January 2024

Abstract

MT systems, as well as other translation technologies, inherit and spread bias from the data they are trained on, reflecting societal bias. Gender bias is by now a primary issue in NLP research, and it is of utmost importance to analyse it in light of the widespread use of these technologies by the general public. Specifically, the use of masculine generics (MG) is one of the most prominent vectors of gender bias in Italian. In this paper, gender bias and MG are discussed in relation to English-Italian translations performed by MT systems. Moreover, the performance of four different models trained on three different data representations is compared in order to train a gender classifier with the aim of identifying Italian translations that use MG when translating English gender-neutral sentences. This is a pilot project which is part of a larger study whose final aim is to train a gender-neutral re-writer for Italian MT output.

1. Introduction

1.1. Gender bias in MT

With the development of machine translation (MT) systems and the fast improvement of their performance, as well as with the availability of free-to-use systems, this technology has become more and more common and accessible to the general public. While this allows for fast and easy communication among people and communities who do not speak the same languages, it is crucial to seriously take into consideration and tackle the problem of bias.

A specific type of bias which is receiving more and more attention – especially in relation to MT – is gender bias. Following the definitions by Savoldi et al. (2021), bias is considered as a skewed representation of reality which results in the systematically unfair treatment of different groups of people. It is inherited by language models from the data they are trained on, which reflect an unfair society (pre-existing bias); moreover, it can be further exacerbated by specific choices made when building the models (technical bias). Specifically, gender bias entails a stereotypical representation of gender roles and the under-representation of women and other groups (including queer and non-binary people).

Gender bias is a particularly relevant problem in MT because different languages have vastly different gender systems, which makes the translation of gender a complex task. Furthermore, gender is a fundamental part of an individual's identity, and for this reason, overlooking this issue not only contributes to the erasure of non-conforming gender identities, but it can also have a significantly negative psychological impact, notably through misgendering (i.e., addressing someone with the wrong gender: see e.g., Lardelli & Gromann, 2023b). Users who are part of groups subject to gender bias are thus more likely to have an unpleasant experience when using MT systems (emerging bias: Savoldi et al., 2021).

Finally, as Gromann et al. (2023) point out, with MT systems being more and more commonly used, their impact is even more widespread since MT output extends to contents that are not explicitly marked as such, which means that users might not be aware they are consuming it. For these reasons, introducing non-binary language in MT systems can have a positive impact on the representation of non-conforming identities.

In the next paragraph, gender bias will be discussed in relation to the problem of translating gender from English to Italian.

1.2. Masculine generics in Italian

While English is considered as a notional gender language, Italian – as most romance languages – is a grammatical gender language (Savoldi et al., 2021). This means that gender in Italian is marked through a set of morphological suffixes which are repeated many times across different parts of speech in the same sentence. On the other hand, English has some gender-specific words (e.g., noun pairs [mother-father], derivational [actor-actress] and compound [chairman-chairwoman] nouns, and pronouns [he-she]), but gender is rarely marked through morphological features.

For this reason, when translating an English sentence with no explicit gender marks or gender-specific words to a grammatical gender language such as Italian, the translator will have to decide which gender to use in the target sentence and use the corresponding gender marks in all the words that are related to the same referent. Most of the time, this decision will be based on context (e.g., according to the gender of the person being referred to, if known). However, the masculine gender is usually the default choice when context is missing (e.g., the referent is not a specific person, or is a group of people, etc.), or even when it is available, in some specific cases (see below); furthermore, when non-binary people are involved, Italian translations tend to use gendered pronouns (masculine or feminine), especially since non-binary issues are still not prominent in the Italian debate around gender bias in language (Lardelli & Gromann, 2023b).

The idea that the masculine gender is the default choice and that it serves as a neutral form results in the use of masculine generics (MG). This term identifies masculine forms used in a "generic" way, to refer to individuals and groups of unknown, irrelevant (i.e., when the referent is not a specific person but rather a general group, e.g., "scientists"), or mixed genders, as well as, in some cases, to specific people who do not identify as men. In particular, the use of masculine titles – which some claim to be neutral (see Gheno, 2020) – to refer to women is at the same time a product and a cause of the marginalisation of women, since it is rooted in and contributes to under-representation (Giorcelli et al., 2015); interestingly, as Gheno (2020) points out, feminine titles are common for professions more strongly associated to women, while MG are the norm when it comes to positions of economic and political power. As a matter of fact, MG are far from being neutral, and cognitively, they are generally perceived as masculine (see Gheno, 2022: 395; Lardelli & Gromann, 2023b: 215).

For all these reasons, gender bias has become a primary topic in MT and NLP research. Some examples of related work are provided in the next section.

1.3. Related work

Through their work based on participatory action research – bringing together stakeholders such as translators, MT experts, and queer people – Gromann et al. (2023) conclude that the ideal solution for gender-fair MT would be a customisable system that can resort to different strategies based on context. This framework challenges the one input – one output paradigm for MT, by recognising the possibility of providing more than one translation for one same source sentence based on and user necessities.

An early experiment in this sense, which inspired this project, is Fairslator[1] (Měchura, 2022), a plug-in for common freely available MT systems (DeepL, Google, Microsoft) that allows the user to intervene and resolve some ambiguities in the source text in order to effectively control different aspects of the MT output. These are the gender of ambiguous referents and the number and formality of second person pronouns (e.g., using plural/formal *vous* or singular/casual *tu* in French to translate the English ambiguous pronoun *you*); once the sentence has been translated by the MT system, a dialog opens where the user can solve the ambiguities detected in the source sentence. When it comes to gender, the user can tick the 'unknown gender' box, and then two different strategies are offered: to use the 'male default' or 'gender-neutral notation' (which is based on gender-inclusive language for French).

Fairlsator is mostly rule-based and does not use machine learning or artificial intelligence. This makes it an interesting project since the rules it is built on can be beneficial to the final performance of the tool. As a matter of fact, Gromann et al. (2023) also suggest incorporating rule-based elements in gender-fair systems to overcome the scarcity of training data as well as to accurately handle specific cases and re-inflection rules.

Moreover, the necessity of introducing non-binary language in MT systems is proved, for example, by Lauscher et al. (2023), who report that most systems produce low-quality output and are prone to misgendering when dealing with neo-pronouns.

Another example that inspired this project is Piergentili et al.'s (2023) work; they release a challenge set for the evaluation of English to Italian translation in terms of gender bias (details are provided in §3), and augment it to train a gender classifier with the same goals.

This project is inspired by the work discussed above as well as by other examples, and it is discussed in more detail in the next section.

1.4. This project

In the context of what described until this point, the aim of this project is to tackle the problem of gender bias and of masculine generics in the Italian language. Specifically, it focuses on the automatic translation of gender-neutral English sentences into Italian, since most MT systems use MG as their default behaviour when translating a gender-neutral sentence. This is a frequent scenario since Italian is a grammatical gender, highly morphological language, where gender is often explicit and every word related to the same referent has to be inflected

[1] https://www.fairslator.com/

according to the referent's gender. As a whole, this project emphasises the importance of improving visibility of all gender identities in texts generated by MT systems; moreover, in this context, MG are considered as a grammatical as well as a translation error, since generally, they do not accurately represent the referents' gender (Giorcelli et al., 2015: 14) and are the result of incorrect assumptions made by MT systems based on the most frequent examples found in their training data (Měchura, 2022). They are thus to be avoided as much as possible.

The creation of the gender classifier described here is a pilot project, part of a larger master's degree final project, with the ultimate goal of creating an automatic re-writer aimed at post-editing automatic Italian translations of gender-neutral English source sentences by avoiding the use of MG and making them gender-neutral (see Lardelli & Gromann, 2023a). The classifier is expected to act as a filter during two different phases:

- firstly, during the collection of a larger training dataset for the re-writer, in order to only collect sentences containing MG;
- secondly, when using the re-writer, in order to filter out sentences that do not need to be converted.

Overall, the goal of the project is to provide better translations of English source sentences that are ambiguous in terms of gender, and ultimately, to contribute to research on gender-fair translation technologies, by mitigating gender bias in MT systems as well as challenging gender binarism and the use of the default masculine in Italian.

More details of the data used to train the classifier discussed in this report are provided in the next section.

2. Data

The classifier was trained on a modified version of the GeNTE corpus[2]. The original dataset contains 1500 sentence triplets following the structure: English source sentence – Italian gendered translation – Italian neutral translation. The original source sentences and gendered translations were collected from the Europarl corpus, while the neutral translations were created by linguists based on specific instructions. 750 English source sentences contained in

[2] https://mt.fbk.eu/gente/

the dataset are gender-neutral, while 375 only contain masculine gender marks, and another 375 only contain feminine gender marks.

To build GeNTE, some modifications to the original translations were carried out to fit the necessities of evaluating the translation of gender by MT systems. Most notably, some of the gendered translations were duplicated and then their referents were converted to the opposite gender (either masculine or feminine) in order to balance the data: for this reason, some of the neutral translations are repeated since the source sentences were not duplicated. Finally, the new, gender-neutral translations were created and added to the dataset. For more details on the creation of this dataset, see Piergentili et al. (2023).

It is important to note that the gender-neutral translations in the dataset adopt a gender-neutral approach or indirect non-binary language (Lardelli & Gromann, 2023b; Savoldi et al., 2021), which means that they do not contain any neo-morphemes or neo-pronouns. Rather, they leverage the epicene words and phrases available in standard Italian, i.e., words that can be grammatically masculine or feminine, but that can be used to refer to human referents of any gender. In this way, the gender-neutral translations contain no explicit gender information just like the source sentences, while avoiding the use emerging strategies that are not (yet) considered as standard nor widely adopted in Italian. Moreover, the only words that differ from the reference (gendered) translations are those that carry gender information.

For these reasons, this dataset was judged ideal for learning the task of recognising sentences that contain MG. Additionally, the type of texts that this collection represents is particularly relevant for the analysis of gender bias and MG (see Piergentili et al., 2023). Administrative and legal texts have a specific importance because:

1. they cater to diverse groups of citizens who must be recognised and represented by their institutions;
2. laws and norms have a direct impact on the lives of the people they are addressed to;
3. on the ground of concerns over clarity and conciseness, they often resort to the use of MG and they are resistant to change (Giorcelli et al., 2015).

Since GeNTE was originally intended as a test set for the evaluation of gender-neutral translation in MT systems, it was further modified to better fit the context of this project. Firstly, as some gendered translations contained some or only feminine referents, these were turned into the masculine gender, in order for all these sentences to only contain MG. Moreover, since the original dataset was artificially augmented as discussed above, most of the neutral translations were repeated: for this reason, all the neutral translations corresponding to originally feminine reference translations (which resulted from duplicated and converted masculine translations) were discarded.

The final dataset used for this study thus contains 1125 masculine, 1125 neutral, and 375 feminine sentences. Since the classifier is only intended to distinguish between sentences that contain MG and sentences that do not, masculine sentences are considered as positive instances (1125 sentences), while neutral and feminine sentences are considered together as negative instances (1500 sentences).

The original dataset in .xlsx format was manually modified to meet the needs of this project, and later processed through a Jupyter notebook in Python, using Google's Colab platform[3]. In order to process the dataset correctly, each sentence was stored in a separate .txt file, and in two separate directories: one for the positive subset, and one for the negative subset. When loading the corpus files, a label was attached to each sentence to identify it as negative (0) – i.e., not containing any MG – or positive (1).

In the following section, the model architectures and dataset representations used to carry out the training of the classifier will be described.

3. Models and representations

3.1. Models

Four different models were trained on three different types of dataset representations (§3.2), in order to determine which configuration led to the best performance.

The tested architectures include two statistical models, namely, a Multinomial Naïve Bayes model (MNB) and a linear Support Vector Machine (SVM); and two neural networks (NNs), namely, a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) network. These models were trained on Term Frequency - Inverse Document Frequency (TF-IDF) vectors and on Word2Vec (W2V) and fastText (FT) embeddings. All the models were built, trained, and tested through a Jupyter notebook as already mentioned, mainly using Sci-kit Learn[4] for statistical models and Keras[5] for NNs.

[3] https://colab.google/
[4] https://scikit-learn.org/stable/about.html; see Pedregosa et al. (2011)
[5] https://keras.io/about/

Table 1 shows the basic architectures for the two NNs. Other hyperparameters were automatically tuned as explained further below.

Model (Sequential())	**Layers**	**compile() method parameters**
CNN	• Convolutional layer (padding='same', strides=1, activation='relu') • GlobalMaxPooling1D • hidden Dense layer (300) • Dropout layer • Activation layer (sigmoid) • output Dense layer (1)	• optimizer algorithm: Adam • loss algorithm: binary cross entropy
LSTM	• LSTM layer • Dropout layer • Flatten layer • output Dense layer (1)	• optimizer algorithm: RMSprop • loss algorithm: binary cross entropy

Table 1: NNs architecture summary

The NNs were built and trained through KerasTuner[6] in order to automatically tune and find the best hyperparameters using the RandomSearch algorithm. A specific tuner was created for each model configuration and run for 10 trials; finally, the best model and hyperparameters were retrieved based on the lowest validation loss reached by that configuration. Each trial was set to train the model for 10 epochs, but early stopping was applied to stop training if the validation loss stopped decreasing for 2 consecutive epochs, and then restore the weights at the best epoch. After finding the best model for each configuration, some models were showing evidence of over-fitting; for this reason, they were re-trained from scratch, thus discarding the best model found by the tuner. The same criteria in terms of epochs and early stopping were followed, while other parameters were changed.

The hyperparameters that were tuned for each NN type are reported in Table 2, along with the range of possible values for each parameter and the best value found by the tuner, as well as the manually set values for models that were eventually built and trained from scratch.

[6] https://keras.io/keras_tuner/

Model type	Hyperparameter	Range	Representation	Best value	Manually set value
CNN	**Filters**	100-500	**TF-IDF**	350	N/A
			W2V	100	N/A
			FT	250	N/A
	Kernel size	3-7	**TF-IDF**	5	N/A
			W2V	4	N/A
			FT	3	N/A
	Dropout	0.1-0.5	**TF-IDF**	0.3	N/A
			W2V	0.4	N/A
			FT	0.4	N/A
	Learning rate	0.000001-0.01	**TF-IDF**	0.001	N/A
			W2V	0.001	N/A
			FT	0.01	N/A
	Batch size	16, 32	**TF-IDF**	16	N/A
			W2V	16	N/A
			FT	16	N/A
LSTM	**Units**	50-250	**TF-IDF**	50	50
			W2V	120	50
			FT	250	50
	Dropout	0.1-0.5	**TF-IDF**	0.2	0.5
			W2V	0.4	0.2
			FT	0.3	0.2
	Learning rate	0.000001-0.01	**TF-IDF**	0.01	0.001 (default)
			W2V	0.001	0.001 (default)
			FT	0.0001	0.001 (default)
	Batch size	16, 32	**TF-IDF**	16	32
			W2V	16	32
			FT	32	32

Table 2: Tuned hyperparameters for each NN architecture

3.2. Dataset representations

As mentioned above, the models were trained on three different representations of the dataset, including one word count representation (TF-IDF) and two different word embeddings

(Word2Vec and fastText). The TF-IDF vectors were created through an original Python function, while the Italian Word2Vec and fastText pre-trained embeddings were loaded and employed through gensim[7].

TF-IDF vectors represent the co-occurrences of words in a document or sentence. The TF-IDF weight calculated for each token is the product of its token frequency (i.e., the number of times it appears in the document) and its inverse document frequency (i.e., the number of documents containing it divided by the total number of documents in the corpus). In this project, TF-IDF scores were calculated at the sentence-level. Moreover, two different data structures were used based on the type of model:

- for the statistical models, a 2D array was used, featuring a 1D vector of token TF-IDF weights for each individual sentence (where each token is represented by a single float);
- for the NNs, a 3D array was created instead, where each token is represented by a n-dimensional vector containing its TF-IDF weights for all sentences; each word vector was then padded or truncated based on the average number of features.

On the other hand, Word2Vec (Mikolov et al., 2013) and fastText (Grave et al., 2018) are algorithms that create word embeddings representing semantic relations among the words in a text corpus; while Google's Word2Vec vectors are only available for English, fastText provides pre-trained vectors for 157 languages. For this study, the Italian Word2Vec pre-trained embeddings[8] developed by Di Gennaro et al. (2021) and the fastText pre-trained embeddings for Italian[9] were used.

Two different data structures were created also for word embeddings, following the same principle described above for TF-IDF vectors:

- for the NNs, the default 3D W2V/FT vectors were used, where each token is represented by a 300-dimensional vector;
- for the statistical models, the datasets were converted into 2D arrays, where each token in a sentence is represented by the average feature of the W2V/FT vector for that token (each token is thus represented by a single float).

Input lengths (maxlen) were set as the length of the longest vector for each representation.

[7] https://radimrehurek.com/gensim/index.html; see Řehůřek & Sojka (2010)
[8] https://mlunicampania.gitlab.io/italian-word2vec/
[9] https://fasttext.cc/docs/en/crawl-vectors.html

Table 3 summarises the vector shape for each representation. In this respect, it is important to note that when using fastText embeddings, there are actually no out-of-vocabulary (OOV) words, since the embeddings for words that do not appear in the pre-trained vectors are created based on embeddings for character n-grams (Grave et al., 2018). As it appears in Table 3, this resulted in longer sequences with fastText vectors when compared to Word2Vec, since for the latter, unknown words were discarded. For reference, the longest sentence in the dataset has 182 tokens. Each dataset contains one vector for each sentence in the corpus, i.e., 2625 vectors.

Representation	Maxlen	Embedding dimensions (for 3D arrays)	Vocabulary (tokens)
TF-IDF	72	373	6,438
Word2Vec	161	300	618,224
fastText	182	300	2,000,000

Table 3: Summary of dataset representation features

The third dimension (embedding dimensions or word vector length) only applies to NNs. As explained above, statistical models were trained on simpler data structures where vectors have as many features as there are unique words in the dataset, i.e., 7996 (see the attached notebook).

The performance reached by each model architecture – dataset representation configuration is reported and discussed in §4 and §5, respectively.

4. Results

Table 4 shows the two statistical models' performance on the test set for each dataset representation. For each metric, the weighted average is shown; bold and underlined figures indicate the best performance for one type of model, while red characters identify the best performance overall (statistical and neural models are considered separately). More details can be found in the notebook, and performance on each class is reported in the confusion matrices further below (Tables 6-9).

Model	**Precision**	**Recall**	**F1**	**Accuracy**
NB TF-IDF	**0.61**	**0.59**	**0.52**	**0.5904**
NB Word2Vec	0.30	0.55	0.39	0.5504
NB fastText	0.30	0.55	0.39	0.5504
SVM TF-IDF	**0.78**	**0.77**	**0.76**	**0.7676**
SVM Word2Vec	0.30	0.55	0.39	0.5504
SVM fastText	0.30	0.55	0.39	0.5504

Table 4: Summary of statistical models' performance on test set for each representation

Table 5 summarises each NN's performance on the test set at its best epoch, i.e., after tuning and training with KerasTuner or after manual training where applicable. All the metrics are computed at the best epoch of the best model, and figures are shown as the weighted average for that metric; the specific metrics for each class can be found in the attached notebook. Bold and underlined figures indicate the best performance in one type of model, while red characters identify the best performance overall (statistical and neural models are considered separately).

Model	**Re-trained**	**Best epoch**	**Precision**	**Recall**	**F1**	**Accuracy**	**Loss**
CNN TF-IDF	no	N/A[10]	0.72	0.72	0.72	0.7180	0.5183
CNN Word2Vec	no	N/A[10]	**0.90**	**0.90**	**0.90**	**0.9009**	0.2551

[10] It was not possible to retrieve the number of epochs for the models trained with KerasTuner.

CNN fastText	no	N/A[10]	0.89	0.89	0.89	0.8876	**0.2451**
LSTM TF-IDF	yes	10	0.72	0.71	0.71	0.7123	0.5179
LSTM Word2Vec	yes	3	**0.79**	**0.78**	**0.77**	**0.7771**	**0.4436**
LSTM fastText	yes	6	0.76	0.76	0.76	0.7638	0.4646

Table 5: Summary of NNs' performance on test set for each representation

As mentioned, the above figures do not explicitly state the different performance of each model in relation to each class in the dataset. To account for this difference – which is quite high for some models, as discussed in §5 – confusion matrices (Tables 6-9) show the proportion of correct and wrong predictions for each class based on the gold standard, calculated at the best epoch of the best model. More details are available in the attached notebook.

NB_TF-IDF	Model predictions	
Gold standard	**0**	**1**
0	266	23
1	192	44
Total	458	67

NB_W2V	Model predictions	
Gold standard	**0**	**1**
0	289	0
1	236	0
Total	525	0

NB_FT	Model predictions	
Gold standard	**0**	**1**
0	289	0
1	236	0
Total	525	0

Table 6: Confusion matrices for NB models with each representation

SVM_TF-IDF	Model predictions	
Gold standard	**0**	**1**
0	260	29
1	93	143
Total	353	172

SVM_W2V	Model predictions	
Gold standard	**0**	**1**
0	289	0
1	236	0
Total	525	0

SVM_FT	Model predictions	
Gold standard	**0**	**1**
0	289	0
1	236	0
Total	525	0

Table 7: Confusion matrices for SVM models with each representation

CNN_TF-IDF	Model predictions	
Gold standard	**0**	**1**
0	229	60
1	88	148
Total	317	208

CNN_W2V	Model predictions	
Gold standard	**0**	**1**
0	261	28
1	24	212
Total	285	240

CNN_FT	Model predictions	
Gold standard	**0**	**1**
0	254	35
1	24	212
Total	278	247

Table 8: Confusion matrices for CNNs with each representation

LSTM_TF-IDF	Model predictions	
Gold standard	**0**	**1**
0	199	90
1	61	175
Total	260	265

LSTM_W2V	Model predictions	
Gold standard	**0**	**1**
0	265	24
1	93	143
Total	358	167

LSTM_FT	Model predictions	
Gold standard	**0**	**1**
0	230	59
1	65	171
Total	295	230

Table 9: Confusion matrices for LSTM networks with each representation

5. Discussion, limitations & future work

The main problem found in this study is that some models clearly over-generalise to the majority class (i.e., 0). This might be due to class imbalance, even though the small difference in the number of samples (1125 vs. 1500 sentences in positive and negative class, respectively) was not expected to cause such behaviour. To solve this issue, it would be ideal to address class imbalance by equally splitting the dataset so as to have a 50-50 proportion of positive and negative instances in both the training and test sets, or by oversampling the minority class (e.g., by finding more sentences containing MG or by randomly duplicating samples already present in the minority class) (Yu, 2021).

Looking at the confusion matrices, this behaviour can be observed in all statistical models trained with word embeddings, which might be too complex representations for those models. However, this was also the case with all LSTM networks trained with KerasTuner, which is why, eventually, they were all trained from scratch. As shown in Table 2, when training from scratch, most hyperparameters were changed in different ways, which allowed for much better performance. The reason for LSTM networks trained with KerasTuner consistently over-fitting has to be investigated in more depth.

When it comes to the best models, it is clear from Table 5 as well as from the confusion matrices that the best performance is achieved by CNNs overall; as for statistical models, the SVM trained on TF-IDF vectors reached much better performance (validation accuracy: 0.7676) compared to the NB model trained on the same representation (validation accuracy: 0.5904), as it appears from Table 4 and the confusion matrices.

As opposed to statistical models, word embeddings seem to guarantee a better performance for NNs; the best CNN and the best LSTM network were both trained with W2V, even though the performance achieved with FT is very close. This result is contrary to expectations: since FT embeddings allow to create vectors for all tokens, while W2V embeddings discard unknown tokens, the former were expected to result in better performance.

The best model overall is the CNN trained on W2V embeddings (validation accuracy: 0.9009). It reaches the best performance in all metrics, except for validation loss (0.2551), which is slightly lower for the CNN trained on FT (0.2451), whose accuracy however stops at 0.8876. When looking at confusion matrices, it appears that CNNs make the less wrong predictions overall; however, the most balanced predictions are those performed by the LSTM network trained on TF-IDF vectors, with a difference of only 5 predictions between the two classes (see Table 9).

Another aspect to be explored would be the optimal input length (maxlen), which was set as the length of the longest vector. As found by Yu (2021: 7) on a similar project, longer vectors seem to be an important factor for achieving better performance with CNNs; however, according to her study, "the best-performing maxlen is [generally] not the mean or max sample length, but the number of tokens in the majority of samples".

Finally, the models' performance on real-world data still has to be analysed. Specifically, since the MG sentences used to train the models in this project (after editing the GeNTE dataset as discussed in §2) only contain masculine gender marks, the models might not be able to correctly identify sentences containing more than one set of gender marks, which might be a common situation in real-world conditions. This limitation is due to the fact that the data used in this project were not meant for this specific use case, but rather, to test MT systems' performance in a controlled framework (Piergentili et al., 2023). It would thus be ideal to re-train the models on more complex data reflecting real-world conditions.

6. Conclusion

In this project, the identification of masculine generics (MG) in Italian was approached as a binary classification problem, where the positive instances are sentences containing MG and the negative instances are sentences that do not contain any (i.e., where human referents are addressed through gender-neutral words or feminine gender marks).

A gender classifier was trained to recognise sentences containing MG. To this aim, four different models (Naïve Bayes, SVM, CNN, LSTM network) were trained on three different dataset representations (TF-IDF vectors and Word2Vec and fastText embeddings). After several experiments, the model reaching the best overall performance on the test set was a CNN trained on Word2Vec embeddings.

This is a pilot project and several limitations to the study and its outcome were discussed. Most importantly, the models' performance has yet to be tested on real-world conditions.

Attached resources

The following resources are attached to this report for reference:

- a Jupyter notebook that was used to:
    - pre-process the dataset and collect data;
    - create dataset representations;
    - train and evaluate models.
- the original GeNTE dataset;
- the modified dataset that was used for this project;
- the best model (CNN trained on W2V) in .keras format (containing its configuration and weights).

References

Di Gennaro, G., A. Buonanno, A. Di Girolamo, A. Ospedale, F. Palmieri, G. Fedele, A. Esposito, M. Faundez, F. Morabito, E. Pasero (2021), ‘An analysis of Word2Vec for the Italian language’, in *Progresses in artificial intelligence and neural systems*, Singapore: Springer, pp. 137-146.

Gheno, V. (2020), ‘La questione dei nomi delle professioni al femminile una volta per tutte’, *Valigia Blu*, 10.12.20, https://www.valigiablu.it/professioni-nomi-femminili/ (retrieved 24.01.2024).

Gheno, V. (2022), ‘Questione di privilegi: come il linguaggio ampio può contribuire ad ampliare gli orizzonti mentali’, *About Gender*, 11(21), pp. 388-406.

Giorcelli, S., M. Spanò, R. Raus, M. Abouyaala, I. Catrano, V. Patti (2015), ‘Un approccio di genere al linguaggio amministrativo’, Università degli Studi di Torino, https://www.unito.it/sites/default/files/linee_guida_approccio_genere.pdf (retrieved 25.01.2024).

Grave, E., P. Bojanowski, P. Gupta, A. Joulin, T. Mikolov (2018), ‘Learning word vectors for 157 languages’, in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, https://aclanthology.org/volumes/L18-1/ (retrieved 24.01.2024).

Gromann, D., M. Lardelli, K. Spiel, S. Burtshcer, L. D. Klausner, A. Mettinger, I. Miladinovic, S. Schefer-Wenzl, D. Duh, K. Bühn (2023), ‘Participatory Research as a Path to Community-Informed, Gender-Fair Machine Translation’, in *Proceedings of the First Workshop on Gender-Inclusive Translation Technologies* (GITT 2023), Tampere, June 2023, pp. 49-59.

Lardelli, M. & D. Gromann (2023a), ‘Gender-fair post-editing: A case study beyond the binary’, in *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, Tempere, June 2023, pp. 251-260.

Lardelli, M. & D. Gromann (2023b), ‘Translating non-binary coming-out reports: Gender-fair language strategies and use in news articles’, *Journal of Specialised Translation*, 40, pp. 213-240.

Lauscher, A., D. Nozza, A. Crowley, E. Miltersen, D. Hovy (2023), ‘What about *em*? How Commercial Machine Translation Fails to Handle (Neo-)Pronouns’, arXiv pre-print 2305.16051v1, https://arxiv.org/abs/2305.16051 (retrieved 25.01.2024).

Měchura, M. (2022), ‘Introducing Fairslator: a machine translation bias removal tool’, in *Proceedings of Translating and the Computer 44*, Genève: Tradulex, pp. 90-95, https://www.tradulex.com/en/pages/Editions-Tradulex-en (retrieved 25.01.2024).

Mikolov, T., K. Chen, G. Corrado, J. Dean (2013), ‘Efficient Estimation of Word Representations in Vector Space’, arXiv preprint 1301.3781v3, https://arxiv.org/pdf/1301.3781.pdf (retrieved 24.01.2024).

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay (2011), ‘Scikit-learn: Machine Learning in Python’, *Journal of Machine Learning Research*, 12, pp. 2825-2830.

Piergentili, A., D. Fucci, B. Savoldi, L. Bentivogli, M. Negri (2023), ‘Gender Neutralization for an Inclusive Machine Translation: from Theoretical Foundations to Open Challenges’, arXiv preprint 2301.10075, http://arxiv.org/abs/2301.10075 (retrieved 18.12.2024).

Řehůřek, R. & P. Sojka (2010), ‘Software Framework for Topic Modelling with Large Corpora’, in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta: ELRA, pp. 45-50.

Savoldi, B., M. Gaido, L. Bentivogli, M. Negri, M. Turchi (2021), ‘Gender Bias in Machine Translation’, in *Transactions of the Association for Computational Linguistics*, 9, pp. 845-874.

Yu, X. (2021), ‘Classifying an Imbalanced Dataset with CNN, RNN, and LSTM’, unpublished project report for the Computational Linguistics course (A.Y. 2020/21), Dipartimento di Interpretazione e Traduzione, Alma Mater Studiorum – Università di Bologna.