Gendered Pronoun Resolution

Human Languages Technologies
Project Report

Gaspare Ferraro, Lucio Messina, Simone Spagnoli

Master Degree in Computer Science - University of Pisa

March 13, 2020

Contents

1 Introduction

In this report we describe our implementation of a variety of different models for **Coreference Resolution**, a task in the framework of the Natural Language Understanding which general form consists in identifying all the mentions that refers to the same entity in a text written in natural language.

Coreference resolution is an important task in natural language processing, indeed up on that is possible to solve higher level natural language processing task that involve language understanding like question answering, information extraction, text summarization and so on. In order to narrow down the scope of our project we select a public Coreference resolution task hosted in a kaggle competition[1] in which the goal was to design a model that, given a span of text containing a pronoun P and two candidate person names A and B, is able to tell whether the pronoun references to A or to B (or Neither of them).

With the idea that we do not have to reinvent the well we plan to build a model made from the fusion and refinement of already existing models. We start looking and analysing the different model from the more innovative to the more classic one, selecting the one with a good trade-off between novelty and diversity. Once we arrive with a decent amount of different models we ensembled them using different aggregation rules. We tried to select different models with high variance in terms of different architecture and performances such that we could improve their results as much as possible using the ensemble technique. We further explore multiple combination of aggregation rules in order not only to improve the final accuracy of the model, but also to discover insight regardless the model itself.

In section 2 we give a more in brief description of the task and the datasets. In section 3 we will explain in more detail the ensemble technique used. In section 4 all the used models are described, and section 5 we report our experiments. Finally in sections 6 and 7 we analyse the result and make some conclusions.

1.1 Bert

All of the analysed models use *BERT* embeddings. The BERT architecture is composed of a sequence of layers of different encoders and it can be trained with arbitrary spans of text to learn a representation of its input. The `BASE` version of BERT was pre-trained on the BooksCorpus and on the Wikipedia Corpus (cased and uncased) and it can be used "as it is" or fine-tuned to solve many specific tasks (question answering, hypothesis-premise pairs in entailment, machine translation and others). The layers or a composition of them can be used as *contextualised word embeddings* that are extensions of the standard word embeddings[2] in which the representation of each word is different depending on the context from which that word was extracted. For a complete description of the Bert architecture refer to Devlin et al. (2018).

[1] `https://www.kaggle.com/c/gendered-pronoun-resolution`

[2] see for example Pennington et al. (2014)

2 Competition

2.1 Task

In a coreference resolution task the aim is to identify spans in a text that refer to the same entity. In this project we focus on a specific sub-domain of coreference resolution that is resolve ambiguous pronoun in English. Recent studies have highlighted that the state-of-the-art coreference resolution systems exhibit gender bias (Webster et al., 2018) (Zhao et al., 2018). An example of a coreference problem is visible in Figure1. To encourage the construction of gender-fair system (Webster et al., 2018) released a dataset with a balanced number of female and male examples on the pronouns resolution task. From this dataset a shared task was published on kaggle[3].

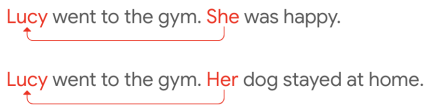

Figure 1: Ambiguous pronoun Example

The kaggle task involve the classification of a specific ambiguous pronoun P in a given Wikipedia span of text (generally a few linguistic sentences) as belonging to one out of three classes: it refers to a given first candidate (A); it refers to a second candidate (B) or neither of them (N). For example : *"John entered the room and saw [A] Julia. [Pronoun] She was talking to [B] Mary Hendriks and looked so extremely gorgeous that John was stunned and couldn't say a word."* here the first candidate is *Julia*, the pronoun is *She* and the second candidate is *Mary*; in this case the task is to identify which entity the pronoun She was referring between Julia, Mary or neither of them. The evaluation metric for the competition is the *Multiclass Logarithmic Loss*[4] considering the tree class: A,B,neither. This loss is computed as following:

$$logloss = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij}\log p_{ij} \tag{1}$$

where n is the number of samples in the test set, m is 3, y_{ij} is 1 if observation i belongs to class j and 0 otherwise, and p_{ij} is the predicted probability that observation i belongs to class j.

The predicted probabilities are not required to sum to one because they are rescaled prior to being scored (each row is divided by the row sum). In order to avoid the extremes of the *log* function, predicted probabilities are replaced with $max(min(p, 1-10^{-15}), 10^{-15})$.

[3] `https://www.kaggle.com/c/gendered-pronoun-resolution`
[4] `https://www.kaggle.com/c/gendered-pronoun-resolution/overview/evaluation`

Even if it was not required by the rules of the competition we also evaluated our models using the *Mean Squared loss*. This loss is computed as following:

$$squaredloss = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}(y_{ij} - p_{ij})^2 \tag{2}$$

2.2 Datasets

The GAP dataset (Webster et al., 2018) was used for this competition. It is based on scraped Wikipedia text snippets, in fact each row of the dataset is composed as follow:

1. ID: univocal name of the example
2. Text: a span of text from the wikipedia page
3. Pronoun: Pronoun in examination
4. Pronoun-offset: Numeric offset of the pronoun in the text
5. A: First candidate antecedent
6. A-offset: offset of the first candidate antecedent
7. A-coref: boolean flags indicating whether the pronoun refers to A
8. B: Second candidate antecedent
9. B-offset: offset of the second candidate antecedent
10. B-coref: boolean flags indicating whether the pronoun refers to B
11. URL: URL of the wikipedia page where the text come from.

The Kaggle competition for this shared task was conducted in two stages: for the first stage three datasets were released: a development set and a validation to train and fine-tune the competitors' models and a test set to evaluate their performances. For the second stage of the competition another "private" (blind) test set was released - the columns *A-coref* and *B-coref* were omitted. The competitors were supposed to retrain up to two models using all the first stage data and submit their predictions for the second stage test set. The final score were computed using the *logloss* formula.

	Number of examples	A	B	N	Male	Female
development set	2000	874	925	201	1000	1000
validation set	454	187	205	62	260	194
test set stage 1	2000	918	855	227	1196	804
test set stage 2	760	?	?	?	403	357

Table 1: Datasets size, class distribution and gender pronoun distribution

3 Ensemble learning

Supervised learning algorithms search through a hypothesis space to find a suitable hypothesis able to produce accurate predictions for a particular problem. The hypothesis space contains often hypotheses that are very well-suited for a particular problem, but it may be very difficult to find a good one. The aim of ensembles is to combine multiple hypotheses to form a (hopefully) better hypothesis.

Studies empirically demonstrated that ensembles tend to yield better result when there is a higher diversity among the models (Kuncheva and Whitaker, 2003). Indeed many ensemble method seek to find and promote diversity among the models they combine (Brown et al., 2005). This can bee clearly see if we come to the real live and we consider each model as individual, in this filed the phenomenon is called *Wisdom of the Crowd* and the following citation by Surowiecki(Gee and Hayes, 2011) explains the concept: *"If you ask a large enough groups of diverse and independent people to make a prediction or estimate a probability, the average of those answers, will cancel out errors in individual estimation. Each person's guess, you might say, has two components: information and errors. Subtract the errors, and you're left with the information"* (Alizadeh et al., 2015) As stated before now is clear why to get even more improvement from the ensemble during the selection of the models we chose much as possible independent and different models.

Ensemble methods become very popular in the last few years and their use in Kaggle (kaggle, 2019) completions and similar one has increased considerably. Their intent is to improve the final prediction, decreasing bias and variance when is needed.

This can be done is several ways, some of the most popular are bagging and boosting, that combine several model of the same type, training that on different set of the training or with different weight for the examples. In our case we wanted to combine different kinds of models, and we want to gain the "best" from each, so we combined the predictions of each model using different aggregation rules.

3.1 Aggregation rules

An aggregation rule is a way to compute the ensemble predictions given the constituent predictions: in all the formulas below $P^c(s)$ are the predictions of the ensemble for the class c in the sentence s, $P_i^c(s)$ are the predictions of the constituent model i for the class c in the sentence s (c can be either A, B or N) and n is the number of constituents. Here is how the ensemble predictions for a sentence s are computed using different aggregation rules:

- *mean*: the ensemble predictions for each class is just the mean of the predictions of the constituent:

$$P_M^c(s) = \frac{1}{n} \sum_{i=0}^{n} P_i^c(s) \tag{3}$$

- *min entropy*: the model with lower entropy on its output probability distribution is considered to be the more confident one and its predictions are used as ensemble predictions.

$$P_Y^c(s) = P_{i^*}^c \tag{4}$$

where

$$i^* \coloneqq \arg\min_i - \sum_{c \in \{A,B,N\}} P_i^c(s) \cdot log(P_i^c(s))$$

- *voting*: each constituent is considered to "vote" for the class with higher predicted probability. The ensemble predictions for the classes are assigned proportionally to the number of votes taken.

$$P_V^c(s) = \frac{1}{n} \#\{i | P_i^c(s) = \max_{d \in \{A,B,N\}} (P_i^d(s))\} \tag{5}$$

- *smoothed voting*: similar to voting, but in the ensemble predictions each class gets a default small probability ϵ. The remaining probability $1 - 3\epsilon$ is split among the classes according to the votes.

$$P_S^c = \epsilon + (1 - 3\epsilon) P_V^c(s) \tag{6}$$

where $P_V^c(s)$ is the probability predicted by the *voting* method. We fixed $\epsilon = 0.01$.

4 Models

In this section we present a brief description of three models that we analysed to solve the Coreference Resolution task.

4.1 Resolving Gendered Ambiguous Pronouns with BERT (5th-place)

This model was presented in Ionita et al. (2019); the source code is publicily available [5].

The proposed solution is implemented as a pipeline which includes several subroutines and it's based on BERT word embeddings. We will going to describe each of the different phases.

4.1.1 Extracting BERT-embeddings for named entities A, B, and pronouns

The embeddings are concatenated for entities A, B, and Pronoun taken from Cased and Uncased large BERT models. Interesting thing is that extracting embeddings from intermediate layers (from -4 to -6) worked best for the task.

Point-wise products of embeddings were added for Pronoun and entity A, Pronoun and entity B as well as AB - PP. First of these embedding vectors expresses similarity between pronoun and A, the second one expresses similarity between pronoun and B, the third vector is supposed to represent the extent to which entities A and B are similar to each other but differ from the Pronoun.

4.1.2 Fine-tuning BERT classifier

Apart from extracting embeddings from original BERT models, a fine-tuned BERT classifier for the task were made. By an appropriate changes to the "run_classifier.py" script from Google's repository. 11 preprocessing input data for the BERT input layer included stripping text to 64 symbols, then into 4 segments, running BERT wordpiece for each segment, adding start and end tokens (with truncation if needed) and concatenating segments back together.

4.1.3 Hand-crafted features

Different hand-crafted features were added to trying improve the results, like syntactic roles of entities A, B, and Pronoun (subject, direct object, attribute etc.) extracted with SpaCy and similar tools.

However adding all these features had only minor effect on the quality of pronoun resolution (resulted in a 0.01 decrease in log-arithmetic loss when measured on the Kaggle test dataset).

[5] `https://github.com/Yorko/gender-unbiased_BERT-based_pronoun_resolution`

4.1.4 Neural network architectures

The final neural network setup includes:

- 6 independently trained fine-tuned BERT classifiers with preprocessing described in 4.1.2
- 5 multi-layered perceptrons trained with different combinations of BERT embeddings for A, B, Pronoun (see 4.1.1) and handcrafted features (see 4.1.3), all together referred to as "frozen" model.
- Blending involves taking predicted probabilities for A, B and "Neither" with weight 0.65 for the "fine-tuned" model and summing the result with 0.35 times corresponding probabilities output by the "frozen" model.

4.1.5 Correcting mislabelled instances

During the competition, more than 150 label corrections were proposed for the GAP datasets[6] when Pronoun is said to mention A but actually mentions B and vice versa. For the GAP test set, this resulted in 66 pronoun co-references being corrected. It's important to mention that the observed mislabeling is a bit biased against female pronouns (39 mislabeled feminine pronouns versus 27 mislabeled masculine ones).

4.2 Anonymized BERT: An Augmentation Approach (7th-place)

The model was presented in Liu (2019); its source code is available online [7]. Bo Liu addressed the gender bias problem by anonymizing all the GAP input datasets. For each sentence in each dataset, the names of the coreference candidates A and B are replaced by two same-gender common English names such as (Alice, Kate) or (John, Michael). A male pair is used if the pronoun P is masculine, a female pair is used otherwise.

The anonymized datasets are used along with the original one in the subsequent phases; the aims of the anonymization process are:

1. Remove any gender bias.
2. Remove idiosyncratic information embedded in individual names.
3. Remove noise due to unseen names during the inference phase.
4. Prevent the word tokenizer from splitting long or uncommon names into different tokens.
5. Get a performances boost by ensembling the results over the five different datasets.

[6] `https://www.kaggle.com/c/gendered-pronoun-resolution/discussion/81331`
[7] `https://github.com/boliu61/gendered-pronoun-resolution`

The final score is obtained by computing the weighted average of the results of two models: the end2end model (weight: 0.9) and the Pure Bert model (weight 0.1). The end2end model has the same architecture described in Lee et al. (2017) with two important modifications: level -4 *BERT* embeddings are used instead of GloVe uncontextualized embeddings; the end2end model does not have mention scores because candidates A and B are given in this task. Some linguistic features are used along with the embeddings to represent the input. The Pure Bert model is a Fully Convolutional Network in which the input is represented by the concatenation of the level -3 and -4 of the *BERT* embeddings of A, B and P.

Bo Liu trained each model both with the Large Uncased Bert dataset and with the Large Cased Bert dataset. They used a 5-fold cross validation to train the four models and to fine tune the ensemble weights for each model: End2end Uncased (weight 0.36), End2end Cased (weight 0.54), Pure Bert Uncased (weight 0.04) and Pure Bert cased (weight 0.06). The overall ensemble placed 7th in the competition.

Although Bo Liu provided a repository with the source code they used to train the model for the competition we were unable to reproduce the results presented in their paper. The main issue was that their Jupiter notebooks were engineered for this very specific task, this makes them difficult to reuse. In particular:

- The datasets filenames and URIs are hard-coded;
- Some pieces of code and data are not included in the repository. They are downloaded "on-the-fly" when the programs are executed.
- The structure of the so called "Drive" folder is not specified.
- The parameters are hard-coded.
- The notebooks make assumptions on the input files length and content.

As specified in their paper Bo Liu used "all of test and development plus 400 random rows in validation set (4400 in total) to train the system and left 54 as a sanity check to test the inference pipeline" while we used the development set only to train other models (see section 2.1). We gave our best effort to standardise the usage of the datasets by modifying their code in order to make it use the development set during the training phase. The Authors of the notebooks suggest to modify some variables (the input filenames and other hardcoded parameters) and run the notebooks again. However, in some parts of the code they just weren't using the value of that variables but reading from other files which names were also hardcoded. Some of these files are copies of the official datasets modified by the authors to correct the wrong labels; other specific errors are corrected by the code itself during the execution. Furthermore, during the preprocessing/anonimization phase the first notebook splits the input datasets in chunks of 1000 sentences each and extracts the features to train the models from each chunk independently. However the other notebooks

expect a fixed number of chunks as input as if they "knew" which dataset was given in input to the first notebook.

We failed in re-engineering Bo Liu's code in order to standardise the usage of the datasets: our version of their model achieves 0.883 logloss on the validation set - even worse than one of our baseline (see section 5.2). However, we applied the techniques described in Liu (2019) to get a performance boost in our other models (see section 5.3).

4.3 Gendered Pronoun Resolution using BERT and an extractive question answering formulation (9th-place)

This model was presented in Chada (2019); the source code is available online [8].

4.3.1 Introduction

The model developed by Rakesh Chada was made by the ensembling of three different models: the question answering (CorefQAExt), multiple choice (Coref-Multi) and sequence classification (CorefSeq) models. For all of the three models was chosen to use the pre-trained BERT embeddings.

4.3.2 Question Answering System (CorefQA)

The model input in this case is in the form of "[CLS] Q [SEP] W [SEP]" where Q represent the question text, and W the Wikipedia text. Q is the pronoun context window of up to 5 words. Each word is tokenized using the SQUAD formulation in Devlin et al. (2018). To calculate the required output probabilities over the given A, B and N, they use a combination of max pooling and logistic regression. Starting from the token logits they extract the span logits by taking the maximum and minimum value corresponding to the offsets of the A and B, plus the max and min logits for the sequence. These six logits are then fed as input to a multi class logistic regression that give us the desired probability p_A, p_B and p_N.

4.3.3 Multiple Choice classification(CorefMulti)

They formulated the task as a SWAG (Zellers et al. (2018)) style multiple choice problem among A, B or N class. For each example they construct four input sequences:

- the given Wikipedia text;
- the concatenation of P followed by the word "is" followed by either A, B or the word "neither"

[8] `https://github.com/rakeshchada/corefqa`

So for example if the wikipedia passage is "they say [A]John and [Pronoun]his wife [B]Carol had a son" the four sequences used as input for SWAG would be

1. They say John and his wife Carol had a son.
2. They say John and his wife Carol had a son. his is John
3. They say John and his wife Carol had a son. his is Carol
4. They say John and his wife Carol had a son. his is neither

4.3.4 Sequence classification (CorefSeq)

In the last model they addressed the problem as a standard sequence classification task. In particular the input of the model is the given Wikipedia page, the sequence features are extracted by concatenating the embeddings corresponding to the tokens A, B and the pronoun spans. The span embedding are calculate by concatenating the token embedding of the start and end token, and the result of an element-wise multiplication of the start and and token embeddings. Each token embeddings come from the output of the last encoder layer of the (fine-tuned) BERT. Finally the token are fed to a single hidden layer feedforward neural network with a ReLU activation, and a softmax layer at the output that provides the predicted probabilities for the three output classes.

5 Experiments

We first set up two very simple models (referred as *modelCosine* and *modelSupervised*) based two naive approaches and used them as baselines to check the quality of the results of the other more complex models. We then considered the three models described in section 4. We programmed three Python wrapper classes in order to standardise the code provided by the three authors by implementing the same interface that can be used to train and evaluate all the models on different datasets. The original core code that trains the models and computes the predictions was not modified[9].

As pointed out in section 4.2 we had problem in adapting the source code provided by Bo Liu, so we did not include their model in our experiments. However, we used a method they describe to preprocess the datasets, generating four anonymized versions of each input dataset.

We used the datasets released for the first stage of the competition for the model selection: we trained five different copies of the model by Ken Kringe (described in section 4.1 and henceforth referred as *model5*) and five different copies of the model by Rakesh Chada (described in section 4.3 and henceforth referred as *model9*). An instance of each model was trained on the original datasets while the other eight instances were trained on the different anonymized

[9]The source code of our experiments can be found at `https://github.com/burlamix/Gendered-Pronoun-Resolution`

versions of the input datasets. We then evaluated each instance singularly and several combinations of ensemble. We used the multiclass logarithmic loss and the mean squared loss to select the two best models, then we retrained them with the datasets released for the second stage and reevaluated them to compute the final scores.

5.1 Physical hardware

The experiments were executed on a server with the following hardware specs:

- Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz (40 cores / 80 threads)
- 512Gb RAM
- 4x Tesla P100 GPU optimised for HPC

All the source codes for the models were highly optimised to exploit the available hardware.

5.2 baselines

We set up two simple models to be used as baselines: both of them use BERT embeddings (see section 1.1) in different ways.

modelCosine is based on the cosine similarity[10]. Two metrics are computed: s_a is the cosine similarity between the embedding for candidate A and the embedding for the pronoun P and s_b is the cosine similarity between the embedding for candidate B and the embedding for the pronoun P. The predicted probabilities for A, B, N are computed proportionally to s_a and s_b.

modelSupervised uses a simple multilayer artificial neural network to solve the coreference resolution problem as a supervised end-to-end task. The input for the neural network is the concatenation of the embeddings for P, A, B, as well as the pointwise products of the embeddings $P \cdot A$ and $P \cdot B$ representing the similarity between the pronoun and each candidate and $A \cdot B - P \cdot P$ representing the extent to which entities A and B are similar to each other but differ from the pronoun[11]. The neural network has 2 dense layers composed of 512 and 256 units followed by a softmax layer of three units that output the predicted probabilities for the three classes.

Apart from the pronoun and the two candidates, the embeddings for the other words of the sentences is not used by the baselines.

[10] see Singhal (2001)

[11] as suggested by Ionita et al. (2019)

5.3 Datasets Anonymization

We applied the technique described in Liu (2019) to anonymize the datasets to be given in input to *model5* and *model9*. It can be used on both labelled and unlabelled datasets as it only requires to know which are the pronoun and the two coreference candidates A and B. For each sentence in the dataset, every occurrence of the names A and B is replaced by two same-gender common English names. A male pair is used if the Pronoun is masculine; a female pair is used otherwise. Four sets of placeholder names are used:

- (Alice,Kate) and (John,Michael)
- (Elizabeth, Mary) and (James, Henry)
- (Kate, Elizabeth) and (Michael, James)
- (Mary, Alice) and (Henry, John)

The names were chosen from the most common names in the first stage data. The substitution is not performed on either A or B (or both) if any of these conditions are met:

1. The placeholder already appears in the sentence.
2. The candidate string is composed of two names but one of them appears in the text alone.
3. The candidate string is composed of three names or more.
4. One candidate is a substring of the other.

We anonymized both the training data (development and validation test sets) and the test data for both the stages of the competition. If a model was trained on a dataset that was anonymized with a placeholder set it has been evaluated with test data anonymized with the very same placeholder set. This ensures that (almost) all the names in the test datasets were already seen by the models during the training phase. Furthermore, in this way we reduced the risk that long or uncommon names were split by the word tokenizers.

Another advantage in using anonymization is that any idiosyncratic information embedded in individual names has been removed by replacing the original names with very common ones. This holds for both male and female names, hence the datasets anonymization should help in removing the gender bias.

Most anonymized models had a considerable performance improvement with respect to the original ones both in terms of logloss and in term of mean squared loss: during the model selection phase the anonymized versions of *model5* performed better than the original one in three cases out of four; the anonymized version of *model9* performed better than the original one in two cases out of four. In the other cases the anonymized versions performed slightly worse than the original ones.

5.4 Models Ensembling

We trained ten different instances of our models with the first stage datasets and we evaluated their performances both as singular models and combining them in different ensembles. The single models performed slightly worse than the performances described in the original papers but significantly better than the two baselines described in section 5.2. Here is a the composition of each ensemble:

- **model5 + model9** is a basic ensemble composed of two models trained with the original datasets.
- **model5 all** is the ensembling of five instances of *model5*: the one trained with the original datasets plus other four trained with the four anonymized versions of the datasets.
- **model9 all** is the ensembling of five instances of *model9*: the one trained with the original datasets plus other four trained with the four anonymized versions of the datasets.
- **model5 + model9 all** is the ensembling of all the trained instances: one instance of *model5* trained with the original datasets, four instances of *model5* trained with the four anonymized versions of the datasets, one instance of *model9* trained with the original datasets and four instances of *model9* trained with the four anonymized versions of the datasets.

For the model selection phase we trained and evaluated each ensemble using all the aggregation rules described in section 3: *mean*, *min entropy*, *voting*, *smoothed voting*.

6 Results and Risk Evaluation

6.1 Baselines results

As shown in table 2 none of the two baselines achieves good results both in terms of logloss and in term of mean squared loss. They are just used to compare the results of the other models and not taken into account during the model selection phase.

Model name	**logloss**	**squaredloss**
modelCosine	0.94659346	0.56592857
modelSupervised	0.76605188	0.50462949

Table 2: Baselines result. the models are described in section 5.2

6.2 Model Selection results

Generally speaking, the *mean* rule worked better than the other ones: in all the ensembles except for *model5 + model9* the ensemble results are better than the results of each one of the base models being ensembled. The *min entropy* rule is less effective as the results of the ensemble are always something in between the results of the single constituents.

The *voting* rule achieved very high logloss (even greater than 1) because of the contributes of the wrong predicted sentences: when all the base models vote for the same class the *voting* ensembler assigns the score 1 for that class and 0 for the other two classes. Each sentence for which the predicted class is unanimously voted but wrong accounts for an amount of $log(10^{-15}) = -34.539$ to be averaged with the contributes of the other sentences[12]. This is an huge amount if compared, for example, to the contributes of the mispredicted sentences by an ensembler that uses the *mean* rule: a quick analysis on the predictions reported that these ensemblers never assign a probability smaller than 0.01.
The *voting* aggregation rule achieves good results in term of mean squared loss because unlike the logarithm it does not suffer from small values amplification. We designed the *smoothed voting* rule specifically to address the problem described above: its performances are comparable to the performances of the *min entropy* rule.

As shown in table 3 The ensemble *model5 + model9* using any aggregation rule is the worst performing one. The reasons could be that too few base instances are ensembled and that the performances of the constituents are too different: the losses of the ensemble is greater than the loss of the best performing base instance.

Model name	**logloss**	**squaredloss**
model9_original	**0.42522399**	**0.23199089**
model5_original	0.49379015	0.27920243
ensemble(mean)	0.43050036	0.23772285
ensemble(min_entropy)	0.43612257	0.23800566
ensemble(voting)	4.08037752	0.29000000
ensemble(smoothed_voting)	0.63958026	0.28400975

Table 3: *model5 + model9* results. The model is described in section 5.4

As shown in tables 4 and 5 The ensembles *model5 all* and *model9 all* achieve better performances than their respective constituent instances when using the *mean* aggregation rule.

As we expected, the best overall result is achieved by the ensemble *model5 all + model9 all* using the *mean* aggregation rule, followed by the same ensemble using the *smoothed voting* rule (see table 6). We selected these two models and

[12]If the provided probabilities were considered to compute the logloss, every sentence that a *voting* ensemble mispredicts would have accounted for an amount of $log(0) = -\infty$. To avoid the extreme values of the logarithm function in this task the probabilities are considered in the interval $[10^{-15}; 1 - 10^{-15}]$: see the official formula for the *logloss* in section 2.1

Model name	**logloss**	**squaredloss**
model5_original	0.49379015	0.27920243
model5_anonymized1	0.52108592	0.28923985
model5_anonymized2	0.39371633	0.20973101
model5_anonymized3	0.40020832	0.20901603
model5_anonymized4	0.42393384	0.22552379
ensemble(mean)	**0.38450822**	**0.20780380**
ensemble(min_entropy)	0.46705271	0.23639385
ensemble(voting)	2.41562121	0.22547999
ensemble(smoothed_voting)	0.47098224	0.22239205

Table 4: *model5 all* results. The model is described in section 5.4

Model name	**logloss**	**squaredloss**
model9_original	0.42522399	0.23199089
model9_anonymized1	0.43897193	0.23781259
model9_anonymized2	0.44300426	0.23932576
model9_anonymized3	0.40990739	0.22188085
model9_anonymized4	0.40371840	0.21430399
ensemble(mean)	**0.39998723**	**0.21375439**
ensemble(min_entropy)	0.41428008	0.21882732
ensemble(voting)	3.01079665	0.24115999
ensemble(smoothed_voting)	0.51272312	0.23635966

Table 5: *model9 all* results. The model is described in section 5.4

retrained them with the datasets released for the second stage of the competition and evaluated them on the blind test dataset.

Model name	**logloss**	**squaredloss**
model9_original	0.42522399	0.23199089
model9_anonymized1	0.43897193	0.23781259
model9_anonymized2	0.44300426	0.23932576
model9_anonymized3	0.40990739	0.22188085
model9_anonymized4	0.40371840	0.21430399
model5_original	0.49379015	0.27920243
model5_anonymized1	0.52108592	0.28923985
model5_anonymized2	0.39371633	0.20973101
model5_anonymized3	0.40020832	0.20901603
model5_anonymized4	0.42393384	0.22552379
ensemble(mean)	**0.37662980**	**0.2018027**
ensemble(min_entropy)	0.45961324	0.23162947
ensemble(voting)	1.76175359	0.21030999
ensemble(smoothed_voting)	0.41459097	0.20772574

Table 6: *model9 + model5 all* results. The model is described in section 5.4

6.3 Blind dataset evaluation results

The results of *model5 + model9 all* on the second stage datasets are shown in table 7. For the sake of the error analysis we reported both the constituent and the ensemble results even if we had to choose only the ensembles as our final models.

Model name	**logloss**	**squaredloss**
model9_original	0.27733286	0.14831148
model9_anonymized1	0.28340086	0.14654532
model9_anonymized2	0.28554203	0.15373259
model9_anonymized3	0.31568031	0.16849566
model9_anonymized4	0.29989617	0.16197376
model5_original	0.34467295	0.19365826
model5_anonymized1	0.25263090	0.13621684
model5_anonymized2	0.25332960	0.14031785
model5_anonymized3	0.31042771	0.17752875
model5_anonymized4	0.29384071	0.16024615
ensemble(mean)	0.26029763	**0.13737616**
ensemble(smoothed_voting)	**0.26002670**	**0.13522451**

Table 7: *model9 + model5 all* results on the blind test dataset. The model is described in section 5.4

A few things are worth to be noted: both the constituents models and the ensemble perform better when trained with the second stage datasets than when trained with the first stage datasets. This is just because the latter are bigger, as pointed out in section 2.1, allowing the models for better learning and generalisation.

As it happened during model selection, when trained with the original datasets *model9* performs better than *model5*. However, using the anonymized datasets *model5* achieves better results. This was not a complete surprise to us because we knew that *model5* ranked fifth on the official competition while *model9* ranked ninth. The anonymized versions of *model9* achieved a greater loss than the original version: it could be the case that the anonymization process can effectively improve the *model9* performances when it is trained using a certain amount of input examples but it may not be so effective for a larger amount of data.

Unlike what happened during the model selection phase the *smoothed voting* aggregation rule works slightly better than the *mean* aggregation rule. Also, two of the four anonymized versions of *model5* performed better than the ensemble. Select the ensembles as our final models was a risk that we had to take because if we had to choose two models out of the ten constituents we would not have known which one could possibly achieve the best results. Thus we selected the ensemble predictions for the evaluation on the blind test set, expecting to achieve good results with the lowest possible variance.

7 Conclusions

In this report we have seen different techniques to solve the Coreference Resolution task and, from the experiments, we have obtained interesting results that we are going to summarize below.

From section 6.1 we have shown that simpler baselines don't achieves good results to be used in practice and, from the first attempts of ensembling (see table 3 for *model5 + model9* results), we have also shown that using the models as they are does not always improves the results.

Although, we were unable to use the model described in section 4.2 directly for the experiments, we will consider it very useful as the datasets anonymization techniques in section 5.3 allowed us to improve the results of the individual models first (see tables 4 and 5 for the single models improvements) and the ensembled results then.

Of the four ensembling rules used (*mean*, *min entropy*, *voting* and *smoothed voting*) we can see that the *mean* rule improves results almost always. The *min entropy* rule follows the trend of the *mean* rule but without get over it. Finally the *voting* rule always worsens the results, due of the logarithmic loss which degrades the result in case of incorrect classification, this led us to implement the *smoothed voting* rules which solves this problem.

To conclude, using the combinations of datasets anonymization and the ensembling techniques we managed to improve the results of the analysed models.

References

Hosein Alizadeh, Muhammad Yousefnezhad, and Behrouz Minaei Bidgoli. Wisdom of crowds cluster ensemble. *Intelligent Data Analysis*, 19(3):485–503, 2015.

Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

Rakesh Chada. Gendered pronoun resolution using bert and an extractive question answering formulation, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

James Paul Gee and Elisabeth R Hayes. *Language and learning in the digital age*. Routledge, 2011.

Matei Ionita, Yury Kashnitsky, Ken Krige, Vladimir Larin, Atanas Atanasov, and Dennis Logvinenko. Resolving gendered ambiguous pronouns with BERT. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 113–119. Association for Computational Linguistics, August 2019. URL `https://www.aclweb.org/anthology/W19-3817`.

kaggle. Kaggle. `https://www.kaggle.com/`, 2019. Accessed: 2019-09-30.

Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51 (2):181–207, 2003.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution, 2017.

Bo Liu. Anonymized bert: An augmentation approach to the gendered pronoun resolution challenge, 2019.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. ACL. URL `https://www.aclweb.org/anthology/D14-1162`.

Amit Singhal. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *CoRR*, abs/1810.05201, 2018. URL `http://arxiv.org/abs/1810.05201`.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. Swag: A large-scale adversarial dataset for grounded commonsense inference, 2018.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/N18-2003`.