FACT2020: Factuality Identification in Spanish Text

Arturo Collazo, Agustín Rieppi, Tiziana Romani and Guillermo Trinidad

Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay

Abstract

In this article we present our proposal for the FACT (Factuality Analysis and Classification Task) challenge tasks 1 and 2. The objective of task1 is to create a system capable of classifying given events found in Spanish texts. Although we present several approaches, the best performing classifier takes an approach of recurrent neural networks trained with embeddings data about the event word and its surroundings, reporting a F1 macro score of 0.6. For task2, a simple rule-base modeling approach is used, reaching a F1 macro score of 0.84.

Keywords

Factuality classification, Factuality identification, FACT, NLP, Neural networks, Random Forest classifier, Word embeddings

1. Introduction

Some of the main objectives in natural language processing are to read, understand and automatically process human language in a machine, this roughly differs from processing programming languages for example. Since the natural language is often obscure, and the linguistic structure depends on several variables, including slang, regional dialects, social context and more, this makes NLP task not trivial.
FACT@IberLEF2020 [1] is a competition where the main goal is to classify events in Spanish texts, regarding their factuality status; classifying event characteristics as well as events themselves. Being able to tag events could come in handy when analyzing news, reports or text in general.

2. Factuality Classification

2.1. Task1 description

Amongst identifiable characteristics in events, factuality would be whether it is certain or uncertain if an event happened. For us, the classification is divided in 3 categories: certain events that happened (facts), certain events that did not happen (counter-facts), uncertain events

Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020)
EMAIL: arturo.collazo@fing.edu.uy (A. Collazo); agustin.rieppi@fing.edu.uy (A. Rieppi); tiziana.romani@fing.edu.uy (T. Romani); gtrinidad@fing.edu.uy (G. Trinidad)
ORCID:

(undefined). The objective is then to train a classifier that can predict the factuality category for tagged events in a given text. Training texts were obtained from Spanish and Uruguayan media.

2.1.1. Initial Data Preprocessing

The training texts are stored in XML files, so the first step was to extract the text to some format we could work on. A long string was created, containing every sentence from the corpus with marked events. This was later split on individual sentences, including the same sentence as many times as the number of marked events on it, having up to one event represented on each.

2.2. RNN + Word Embeddings

2.2.1. Preprocessing

After having the structure described before, sentences get split into words, and each word is translated to Word Embeddings, making use of an embeddings file trained using word2vec over the corpus from [2]. So words turn out as 300-dimensional vectors. In order to distinguish the event word in the sentence, an extra bit is added to this 300-dimension making it 301. This bit value depends on whether the word is an event or not.

Other approaches included representing words as POS-Tags. This was achieved with nltk [3] Stanford POS-Tagger [4], and vectors were represented using an internal form of vector codification, based on tag classes and attributes. The output had a considerably low accuracy.

2.2.2. Padding

Next step was padding the sentences in order to normalize the length of each input for the neural network. This is done applying as much padding as the longest sentence available.

2.2.3. Class weights

Given the training corpus was extremely unbalanced, weighting classes seemed accurate. The weights got from the training corpus are:
- facts: 0.9121
- counter-facts: 2.1925
- undefined: 11.3884

2.2.4. RNN

For the neural network implementation, TensorFlow [5] Keras library [6] is used. The sequential model, consists of a GRU layer with 200 neurons and a dense layer with a 3-dimensional output, representing each of the possible categories. The model is compiled using a categorical_crossentropy loss function, and adam optimizer.
For choosing the correct amount of epochs, an early stopping we used. We concluded the best amount of epochs was between 25 and 30. So they are set to 28. Similarly some exploratory

testing is done on batch size, after trying different values, 30 is chosen as one of the possible values for batch size.

2.2.5. Results

from this approach are:

- **Precision:** 0.611
- **Recall:** 0.603
- **F1-macro:** 0.607
- **Accuracy:** 0.848

2.3. RNN + char level

This technique is strongly based in Aspie96[7], where the unit of information is the character of an event and its neighbors, including spaces and non word characters.

2.3.1. Preprocessing

First, each event is divided into a single char list, as they left and right neighbor to be concatenated in a greater one, if they fit into the window. Then, that list with the event in its center is encoded on every character with one hot encoding (the representation for each character was retrieved by a dictionary with all possible characters that may appear in the sources). Last, to recognize the characters that were part of the event, all of them were marked with a flag for that purpose.

2.3.2. RNN

For the neural network implementation, TensorFlow Keras library [6] is used. The sequential model, consists in two LSTM layers with 75 units each one with and a dense layer with an 3-dimensional output. Sigmoid as activation function, categorical crossentropy as loss function and Adam as optimizer. Besides, the training has a boundary of 14 epochs without improvement over a maximum of 150 epochs.

2.3.3. Results analysis

In the table below, it can be seen how precision and recall decreases between the datasets, which leads to a decrease of f1 metric as well. The lost of performance could be based in the difference on the datasets structures, and a probable little overfitting on the training stage.

Table 1
Results comparison between datasets.

Metric	Train	Validation
macro-precision	**0.677**	0.556
macro-recall	**0.703**	0.545
macro-f1	**0.689**	0.550
accuracy	0.789	**0.798**

2.3.4. Future work

For future work on this approach, several configurations on the model can be done to achieve better results. These include modifying the window size, and customizing hyper parameters related to the model such as recurrent activation function, number of units in each layer or the optimizer.

2.4. Random Forest with Tag counts

2.4.1. Preprocessing

This approach is based on a morphological analysis of the context on each event. For every event we contemplate the event itself and a fixed window at a backward word level. Then a Part-of-speech tagging (POS Tag) is made to the extracted sentence using the Spacy Spanish POS-tagger[1].

Among a lot of information the tagger returns for each word a label which identifies what kind of POS-tag every word has. With the given information a count of the number of apparitions of every POS-Tag, also it is possible to count more than once the POS-Tag of the event to classify.

Count results are mapped to an array, on which every position represents a POS-Tag. The resulting array will be the input for the classifier.

2.4.2. Random Forest

For this model we use Random Forest from sklearn [8] to classify[2], which receives the described input as an entry, and has as output 3 possible values corresponding to the task.

At the stage of model training, the values of the window length and the times that the event's POS-Tag is counted are tuned. The configuration which gives better results is counting twice the event's POS-Tag and using a window length of two words.

[1]https://spacy.io/models/es
[2]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Table 2
Global results comparison.

Metric	RNN + WE	char RNN	SVC	RandomForest	Baseline
macro-precision	0.611	0.556	**0.620**	0.518	-
macro-recall	**0.603**	0.545	0.574	0.592	-
macro-f1	**0.607**	0.550	0.592	0.542	0.246
accuracy	**0.848**	0.798	0.831	0.797	0.524

2.5. SVC with Tag counts + Word Embeddings

2.5.1. Preprocessing

This approach is an extension of Random Forest with windowed input described before. The main idea is to add more information about the event itself. To achieve this a word embedding representation of the event is concatenated to each input, making use of an an embeddings file trained using word2vec over the corpus from [2] to do the encode.

2.5.2. SVC

The first attempt was using a Random Forest to classify, however the results were not good. The second attempt was with an SVC classifier from Sklearn[3], which gave better results than Random Forest and the approach with Random Forest with windowed input.

For the training the values of the window length and the numbers of times that the event's POS-Tag is counted are tune. The best configuration was using a windows length of two words and counting one time the event's POS-Tag.

2.6. Results

The next table shows metric results over the test data, for each of the models described before. We can observe how the RNN plus Word Embeddings and the SVC approach ended up head to head, with 0.015 difference in F1 metric and 0.017 difference in accuracy. One interesting thing to see is how the SVC's precision was a little higher than the RNN, yet the RNN won over recall. Each of the models had a much higher f1 metric than the baseline project.

3. Event Identification

3.1. Task2 description

This task is the previous step of Task1, aiming to automatically identify events in a given text. The input is plain text and the given algorithm has to output the index of words which represent events in it.

[3]https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

For instance, if the input is *El/1 volcán/2 de/3 Fuego/4 ha/5 vuelto/6 a/7 la/8 normalidad/9 ,/10 aunque/11 mantiene/12 explosiones/13 moderadas/14.* It should output *5, 6, 12, 13.*

3.2. Baseline

The competition organizers proposed a simple algorithm in order to set a baseline for the competitors. This classifiers assigns the class 'event' to the words tagged as 'event' at least once in the training corpus. This approach gets a **F1-score** of **0.597**.

3.3. Verbs detection

There are two types of event, verbal and noun. Studying the training corpus (same used in task1) the team noticed that noun events are just **16.5%** of the total. This motivates a simple rules approach, where the classifier identifies a word as an event if it is a verb.

The code is as simple as described, using nltk Stanford POS-Tagger and checking if it determines that the word to classify is a verb or not.

Three metrics are used for the task evaluation, the results obtained for this approach are:

- **Precision:** 0.993
- **Recall:** 0.736
- **Macro-f1:** 0.845

The high precision is due to the fact that almost every verb is an event, making those predictions trivial. Having a high recall means that the test corpus is also unbalanced and most of the events are verbal, as the team proposed.

3.4. Verbs detection + Nouns detection

In order to include noun events to the classifier and inspired by the baseline approach, this rule is also added to the algorithm, assigning the class of 'event' to verbs and to nouns that appear at least once as events in the training corpus.

This approach beats the previous one, obtaining a higher macro-f1 score, caused by a much better recall:

- **Precision:** 0.950
- **Recall:** 0.792
- **Macro-f1:** 0.864

3.5. Future work

The good results obtained with such a simple approach are promising, due to a lack of time the team could not explore more complex solutions, but it would be interesting to test machine learning techniques to identify more noun events.

Although it is well known that noun events are context dependent, the results obtained in the second approach are proof of this, reducing precision (this means we have more false positives). Using some ideas from Task1, it would be interesting to use windows around the words to classify them. This would give the classifiers the possibility to learn from context, rather than just the words.

4. Conclusions

For the firs task of Factuality Classification four approaches were implemented, the best performing (using the macro-f1 score as reference) is Recurrent Neural Network combined with Word Embeddings, which obtained a macro-f1 score of **0.607**.

The second task of Event identification was attempted with two rules classifiers, due to it's known characteristics. The one that got the best results was based on two rules: word **w** is an event if (and only if) (1) **w** is a verb, or (2) **w** appeared in the training corpus as an event. This classifier obtained a macro-f1 score of **0.864**.

For the latest task, the team believes that the use of context and some more complex techniques could greatly improve the obtained results.

References

[1] A. Rosá, L. Alonso, I. Castellón, L. Chiruzzo, H. Curell, A. Fernández, S. Góngora, M. Malcuori, G. Vázquez, D. Wonsever, Overview of FACT at IberLEF 2020: Events Detection and Classification (2020).

[2] A. Azzinnari, A. Martínez, Representación de Palabras en Espacios de Vectores, Proyecto de grado, Universidad de la República, Uruguay, 2016.

[3] E. Loper, S. Bird, Nltk: the natural language toolkit, arXiv preprint cs/0205028 (2002).

[4] K. Toutanova, D. Klein, C. D. Manning, Y. Singer, Feature-rich part-of-speech tagging with a cyclic dependency network, in: Proceedings of the 2003 conference of the North American chapter of the association for computational linguistics on human language technology-volume 1, Association for Computational Linguistics, 2003, pp. 173–180.

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. URL: http://tensorflow.org/, software available from tensorflow.org.

[6] F. Chollet, Keras, https://github.com/fchollet/keras, 2015.
[7] V. Giudice, Aspie96 at FACT (IberLEF 2019): Factuality Classification in Spanish Texts with Character-Level Convolutional RNN and Tokenization, in: Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019), CEUR Workshop Proceedings, CEUR-WS, Bilbao, Spain, 2019.
[8] F. Pedregosa, 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, Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.