¿QUÉ?: A NEURAL NETWORK'S ASSESSMENT OF THE COLLOCATION BETWEEN NULL COMPLEMENTIZERS AND SPANISH VERBS

Isaac Ang
Leland High School
Mehta+ Tutoring

Olivia Bottomley
The Pennington School
MehtA+ Tutoring

August 4, 2021

ABSTRACT

In a sentence, a null complementizer is a complementizer expected to be present when there is not one. Throughout history, Spanish speakers have steadily been dropping complementizers out of their sentences. Although linguists have attempted to study this phenomenon, the process is tedious. Thus, this paper uses computational power and machine learning in hopes of significantly quickening the procedure. We performed two analyses. First, by consolidating the corpora in a Pandas dataframe, we found that volitional verbs have *que* following them the least often, which agrees with prior linguistic research. Our second analysis utilized a LSTM(Long Short-Term Memory) machine learning model to learn not only the syntax but the semantics of Spanish. Testing on a portion of untrained corpora, our model predicted if *que* followed a verb with 74.25% accuracy.

1 Introduction

Throughout the development of modern language, the usage of *complementizers*, words that link subject and subordinate clauses, has dwindled. Riccelli (2018) [1] investigated a drop in the complementizer 'that' in the English language. He found that *that* appears 98% of the time[1] in ancient texts such as the Wycliffe Sermons(c.1350). In contrast, today, the complementizer *dis*appears 90% of the time in common matrix verbs[2]. This omission of the complementizer creates a *null complementizer*. In this paper, we will analyze the role the verb plays in affecting the likelihood that the Spanish complementizer *que* appears in the sentence.

In essence, the complementizer introduces a description of what we are thinking or saying. The complementizer *que* appears after verbs to introduce the embedded clause, as in "Lamento *que* no estés contenta"(I am sorry that you are not happy). However, if you choose to drop the complementizer("Lamento no estés contenta"), the sentence is still grammatically sound. In other words, *que* is assumed to be present. In the field of linguistics, this sentence would become "Lamento $\emptyset$ no estés contenta", where $\emptyset$ denotes a null complementizer.

Not all verbs have a complementizer following them, and those that do are generally intellectual verbs. Riccelli (2018) [1] split these verbs into three categories: epistemic, volitional, and stative. Epistemic verbs describe attitudes and ways of seeing the world(e.g. to think). Volitional verbs describe desires and hypothetical states(e.g. to wish). Stative verbs describe states of being(e.g. to say). Our first analysis evaluates these three verbs and how often they are followed by *que* in historical corpora.

[1]When grammar expects *that* to appear, it appears 98% of the time

[2]Matrix verbs are the verbs of the matrix clause. In the sentence, "Mary wondered whether Bill would come", "wondered" is the matrix verb. "Mary wondered" is the matrix clause and, "Bill would come" is the embedded clause

2 Related Work

There has been research in the field of null complementizers(as unlikely as it may sound). Riccelli (2018) [1] investigated the variation between null and overt expressions of *que* between two Spanish dialects. Our paper builds on Riccelli's work by encompassing many, as opposed to two, dialects of Spanish. Additionally, Yoon (2015) [2] proposed that the interaction of the verb and the sentence contributed to the likelihood of *que*-drop. Our paper builds upon Yoon's work by creating a model that inherently has learned how their interaction contributes to the likelihood of *que*-drop. Tagliamonte & Smith (2005) [3] conducted a similar study with British English, finding *that*-drop in 91% of the cases *that* was expected. Previous research in this field suggest that volitional verbs are the most likely to precede *que*; this study tests and ultimately confirms that hypothesis.

3 Methodology

3.1 Dataset

Our dataset consists of 1.5 million sentences extracted from the Spanish corpora. These sentences were split into 3 categories based on whether they contained an epistemic, a volitional, or a stative verb.

3.1.1 Preprocessing

File to Text Out of the 14 columns in the original file, we dropped irrelevant columns such as author, title, and genre, leaving behind only the column which had the sentence. We then combined the sentences into a dataframe, creating a column to save verb type for later uses. To help with *que*-detection, we added a column containing the verb-of-interest for every sentence. Finally, we added a column called "Exists", which *que*-detection would later fill out. Figure 1 displays our dataframe at the end of this process.

	CONCORDANCIA	Verb Type	Verb	Exists
0	No, no sentí nada, no me di cuenta.	e	dar(se)cuenta	0
1	En la cámara siguiente, de unos veinticinco me...	e	dar(se)cuenta	0
2	-¿Qué ocurre? ¿Estoy soñando? Percibo vagament...	e	dar(se)cuenta	0
3	La Profesora, mientras escudriña los alrededor...	e	dar(se)cuenta	0
4	Duvúrai casi respira unas facciones que oscila...	e	dar(se)cuenta	0
...	...	...	...	...
1565201	A tal extremo han llegado las cosas que el pre...	s	afirmar	0
1565202	El vicepresidente Rafael Alburquerque inauguró...	s	afirmar	0
1565203	Afirman faltan recursos	s	afirmar	0
1565204	mi apoyo. Ahora, le dije que redactara un docu...	s	afirmar	0
1565205	Afirmó que el número de accidentes se redujo e...	s	afirmar	0

Figure 1: CONCORDANCIA(sentence), Verb Type(type of verb in sentence), Verb(verb-of-interest), Exists(1 if *que* followed a verb and 0 otherwise)

SpaCy *que*-detection In order to evaluate whether *que* followed a verb, we used SpaCy's Lemmatizer. For every sentence, we first lemmatized the words in order to retrieve the infinitive form of every verb. Then, we found the verb-of-interest and checked whether *que* existed within 4 words after verb[3]. If so, "Exists" is set to 1, and if not, 0. The

[3]To ensure the tagger did not find a *que* unrelated to the verb, we set a limit of 4 words after the verb, as that is typically the maximum distance between a verb and its complementizer. In the sentence, "Busco un amigo que sea inteligente"(I look for a friend who is smart), *que* is still relevant, as it refers to the verb "Busco". However, in the sentence, "Busco un amigo y la persona que es inteligente no es mi amigo"(I look for a friend and the person that is smart is not my friend), *que* appears more than 4 words after the verb. Now, it refers to the person, not the verb.

"Exists" column would be critical for validation later on. We also calculated the frequency of *que* showing up after each verb type(see Table 1).

Table 1: *Que*-frequency for verb types

Verb Type	*Que*-Frequency(%)
Epistemic	∼21.71
Volitional	∼17.90
Stative	∼22.74

As per Table 1, volitional verbs experience the most *que*-drop, while epistemic and stative verbs experience similar amounts of *que*-drop.

Text to Tokens The final stage of preprocessing prepared the data for our neural network. We allocated one half of the data to training[4]. For every sentence, we removed punctuation and symbols, lowered the words to minimize vocabulary size, and finally separated out distinct words. We split each sentence into consecutive sequences of 5 words(further explained in 3.2). Finally, we implemented Tokenizer(a pre-made class), which mapped distinct words to integers since the Embedding Layer is compatible only with integers. Figure 2 shows a sample of consecutive 5-word sequences.

```
['el hecho que se convoquen',
 'hecho que se convoquen plazas',
 'que se convoquen plazas masivas',
 'se convoquen plazas masivas abre',
 'convoquen plazas masivas abre las',
 'plazas masivas abre las puertas',
 'masivas abre las puertas a',
 'abre las puertas a los',
 'las puertas a los recién',
 'puertas a los recién titulados']
```

Figure 2: Our model is trained on sequences of 5 words. Each sequence consists of a 4-word long input(feature) sequence and a 1 word output(label).

3.2 Model

We used an LSTM(Long Short-Term Memory) to predict if *que* follows a verb in a sentence. Specifically, we used an Embedding Layer to learn the representation of words and a Long-Short Term Memory, LSTM, to predict *que* based on a group of word embeddings. We trained our model on 5-word sequences, where 4 words are the features and 1 word is the label(see Figure 3).

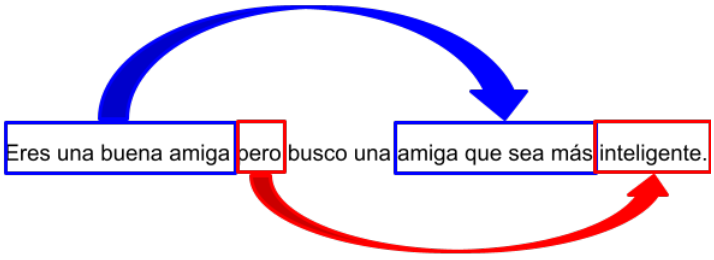

Figure 3: The sentence is split into consecutive 5-word sequences. In each sequence, the model takes in 4 words and attempts to predict the next word. The model trains on the first 5-word sequence, shifts one word to the right, and repeats the same process until the sequence frame moves out of the sentence.

Figure 4 shows the components of our model.

[4]While it is more common to allocate 80% of the data to training, we worked with a giant dataset. Thus, 50% of our dataset was still quite significant. In addition, Google Colab lacked sufficient Random Access Memory to process the 80%

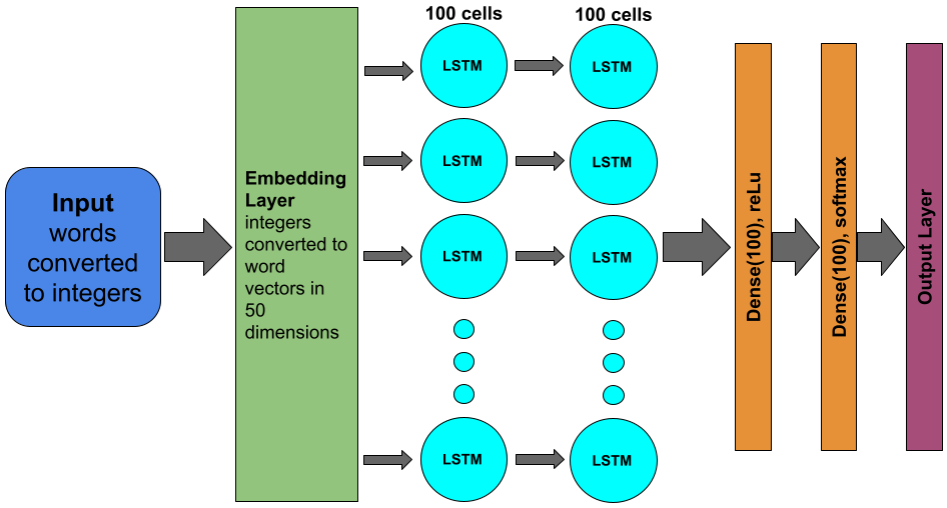

Figure 4: Our model takes an input of one-hot encoded words. The embedding layer converts each word into a word vector in 50 dimensions. Then, 2 100-cell LSTMs predict the next word using the word vectors. The Dense layers fully connect the neurons, changing the dimensions of the vector. Finally, a softmax activation function normalizes the probabilities and the model outputs the most likely word.

The loss function punishes a machine learning model when it deviates too much from the desired results. Our loss function was sparse categorical entropy, as defined by:

$$CCE(p,t) = -\sum_{c=1}^{C} t_{o,c} log(p_{o,c})$$

4 Results

We validated our model by comparing the model prediction against the spacy-tagged column for 1000 sentences not in the training set.

Process For each sentence, we ran the model on all possible 5-word sequences. A sentence has *que* following a verb if any of the sequences have *que* following a verb(see Figure 5). To check whether the model was correct, we compared whether its prediction(0 or 1) was equal to that sentence "Exists" attribute(0 or 1).

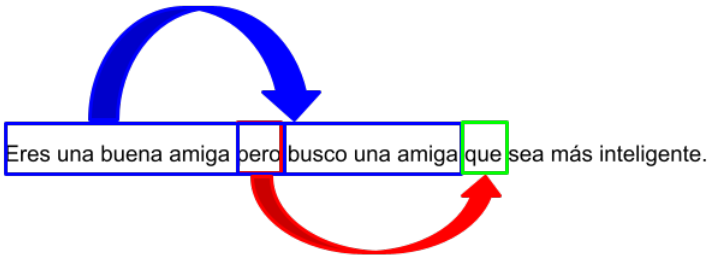

Figure 5: Our model predicts each word of the sentence using the previous 4 words. If, at any point, the model predicts *que* and a verb precedes *que* by a maximum of 4 words, *que* follows a verb for that sentence.

Confusion Matrix For a more accurate representation of our model's accuracy, we utilized a confusion matrix. A confusion matrix has 4 cells: True Positives(TP), True Negatives(TN), False Positives(FP), False Negatives(FN). True predictions are correct and False predictions are incorrect. A Positive prediction occurs when the model predicts that *que* exists and a Negative prediction occurs when the model predicts that *que* does not exist.

Accuracy Our validation accuracy was 74.25%[5]. Our sensitivity(47.5%) was greater than our specificity(26.75%); our model correctly predicted *que* more than it correctly predicted that there wasn't *que*.

5 Conclusion

Obstacles

1. Working with a huge dataset exceeded Google Colab's RAM limits. Consequently, we implemented solutions such as deleting unnecessary data structures from memory and using a sparse model[6].
2. Since our model was not trained to predict verb-related *que*'s, we could not punish the model for predicting noun-related *que*'s as well. Thus, we had to set up another variable - True Neutral - which we incremented whenever the model predicted a noun-related verb. As a result, a lot of False Positives were transferred over to True Neutral, and in the final calculation of validation accuracy, True Neutrals were excluded from the calculation.

Future Work Our paper does not mark the end to the study of null complementizers. In fact, these are some potential improvements to our model:

1. Allow model to predict *que* if it is within the first 4 words of the sentence
2. Test model on tweets to better understand the collocation between null complementizers and verbs in the Spanish language in its modern form
3. Convert all verb forms to the infinitive to minimize vocabulary size when training
4. Use SpaCy's dependency tree to more accurately predict if *que* is dependent on a verb
5. Train model on more epochs
6. Train model just on sequences with *que* as the label to specialize the model

6 Division of Labor

We divided the work as follows:

- Isaac: preprocessing datasets, writing model, training model, writing paper
- Olivia: researching RNNs, finding code documentation, creating poster, proofreading paper

7 Acknowledgements

We would like to thank Ms. Haripriya for making communication possible between students and professors.
We would also like to thank Mr. Bhagirath for offering critical suggestions which improved our model.
We would like to thank Mr. Mohammad and Ms. Andrea for helping debug our code.
Finally, we would like to thank our mentors Dr. Adrian Riccelli and Dr. Colleen Balukas for providing inspiration and structure to our project.

References

[1] Adrián Rodríguez Riccelli. Espero estén todos: The distribution of the null subordinating complementizer in two varieties of spanish. In *Language Variation and Contact-Induced Change*, pages 299–333. John Benjamins, 2018.

[2] Jiyoung Yoon. The grammaticalization of the spanish complement-taking verb without a complementizer. *Journal of Social Sciences*, 11(3):338, 2015.

[3] Sali Tagliamonte and Jennifer Smith. No momentary fancy! the zero 'complementizer'in english dialects. *English Language & Linguistics*, 9(2):289–309, 2005.

[5] (TP + TN)/(Total - TN)

[6] We switched our loss function from categorical cross-entropy to sparse categorical cross-entropy in order to decrease memory usage. The Sparse version was advantageous because it saved only the cells that were filled with data, discarding the rest.