Automatic Syntactic Sentence Simplification with Transformation Based Learning: An Example

Eric Baucom

September 19, 2018

In this short document, we demonstrate a handful of fully worked examples of syntactic sentence simplification within the Transformation-Based Learning (TBL) framework. TBL is trained by operating on a series of templates, iterating through the corpus, creating one rule at a time which reduces the overall error between the training and test sides. The rules are aggregated over the iterations and applied in the order in which they were created. The iterations continue until no more error reduction is possible or until some predetermined threshold of accuracy is achieved. Let us consider the following sentences, first the original "complex" sentence (after hand-curation) and then the simplified sentence:

> In the 17th century the Empire was shattered by the Thirty Years ' War .
>
> In the 17th century the Thirty Years ' War shattered the Empire .

In this example, we are dealing with the phenomenon of *passive sentences*, one of three to be addressed in the thesis (the others being *coordinate phrases* and *quotative sentences*). The task of the simplification system will be to change the passive or "complex" sentence into its active or "simplified" equivalent.

The sentences will first be parsed using a constituency-based parser such as the Berkeley Parser. TBL relies on comparing the labels of tokens from the previous iteration of the algorithm to the current one. Candidate tokens will be collected by the longest matching substring algorithm. We will ensure they are constituent tokens using nltk's "minimum spanning tree" method, which gives the tree position of the smallest subtree which dominates all and only a given string of text. If a candidate token throws errors here, it may indicate a parse error and could be useful in discussion and indicate areas for further research (e.g., especially domain adaptation and other methods to improve parse quality). Many errors may indicate that a full parse is the wrong level of analysis, and it may then behoove us to investigate partial parsing or chunking. We will see that a subroutine will be necessary in coordinated sentences and reported speech sentences in order to correctly resolve some constituents which end in verbs.

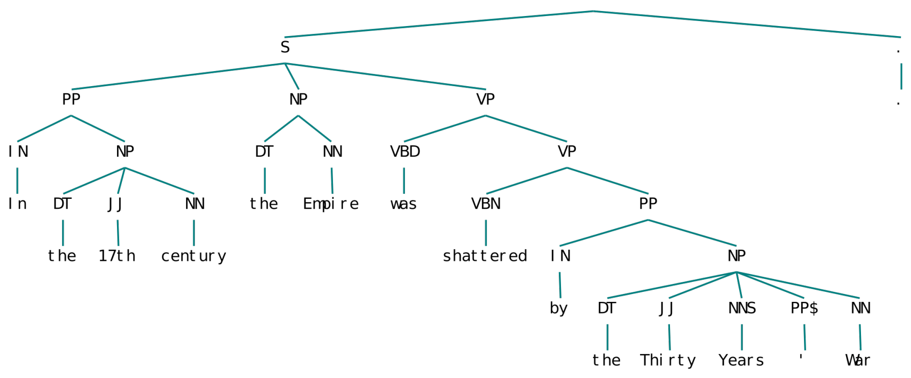

Figure 1: The original complex passive sentence (after hand-curation).

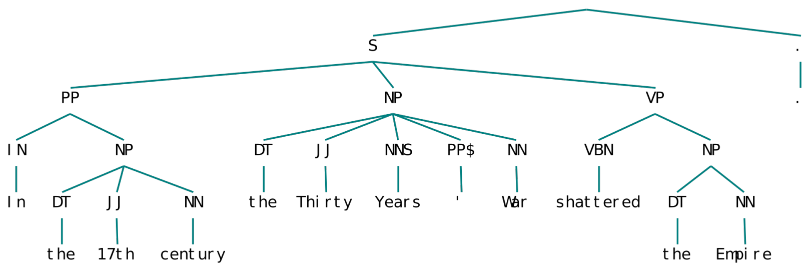

Figure 2: The simplified active sentence.

Following this longest matching substring and minimum spanning tree scheme, the following constituents will be extracted as being in common between the two example sentences:

1. (**PP** `In the 17th century`)
2. (**NP** `the Empire`)
3. (**VBN** `shattered`)
4. (**NP** `the Thirty Years ’ War`)
5. (**.** `.`)

The constituents will then be assigned supertags in both the passive and active sentences which will correspond to their relative positions to the anchor of the sentence. The anchor will be the highest VP node of the sentence, which can be determined algorithmically via a breadth-first search. The highest VP is a useful anchor given its proximity to both the subject (typically left sibling) and the object(s) (typically the children to the right of the main verb child). The convention for supertags will be descriptive in relation to the anchor. The anchor will be represented by the character `a`. The descriptor `up` is used to indicate going to the parent of the current node. The other descriptors will be natural numbers indicating which child node to go to, in left-to-right linear order (it may prove to be useful to allow negative numbers indicating right-to-left linear order, which would mean multiple potential supertags for each token, but we will not discuss that further in this example, as it may or may not prove useful as we scale up to the whole corpus). The supertag (a, 2, 3) would mean the constituent that is the anchor's second child's third child. See below for more examples.

Notice that in the previous list, 1 and 5 maintain the same positions relative to the anchors of the two sentences, so they will be ignored as they present no "errors" to the TBL system (i.e., they have the same supertags in both sentences). The errors for the other consituents are shown below. The convention for errors in our implementation of TBL will be ***<supertag-a, supertag-b, number-of-errors>*** (the number of errors is how TBL judges improvements between consecutive iterations).

2. **<(a,up,2), (a,2), 1>**
3. **<(a,2,1), (a,1), 1>**
4. **<(a,3,2), (a,up,2), 1>**

For an example of coordinated sentences, we consider the two sentences below, first the complex version and then the simplified version.

> April is the fourth month of the year in the Gregorian Calendar ,
> and one of four months with a length of 30 days .

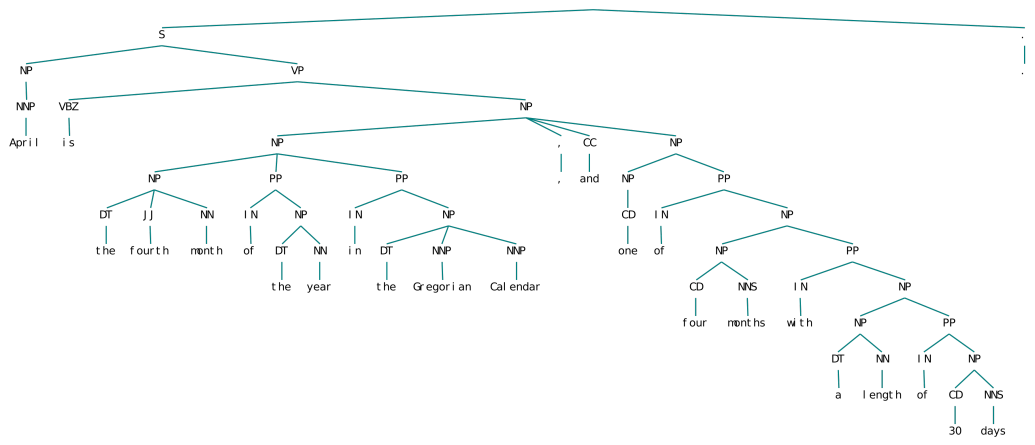

Figure 3: The original complex coordinated sentence (after hand-curation).

> April is the fourth month of the year in the Gregorian Calendar .
> April is one of four months with a length of 30 days .

The following tokens are generated from these sentences:

1. (**NNP** `April`)
2. (**VBZ** `is`)
3. (**NP** `the fourth month of the year in the Gregorian Calendar`)
4. (**NP** `one of four months with a length of 30 days`)
5. (**.** `.`)

For an example of reported speech sentences, we consider the two sentences below, first the complex version and then the simplified version.

> Galen said these elements were used by Hippocrates to describe the human body .

> Galen said X . X is these elements were used by Hippocrates to describe the human body .

The following tokens are generated from these sentences:

1. (**NNP** `Galen`)
2. (**VBD** `said`)
3. (**NN** `X`)

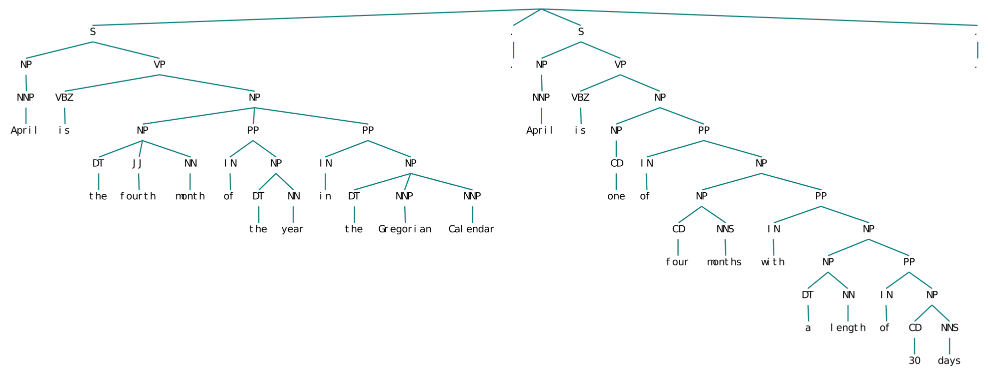

Figure 4: The simplified “de-coordinated” sentence.

4. (**VBZ** `is`)
5. (**S** `these elements were used by Hippocrates to describe the human body`)
6. (**.** .)

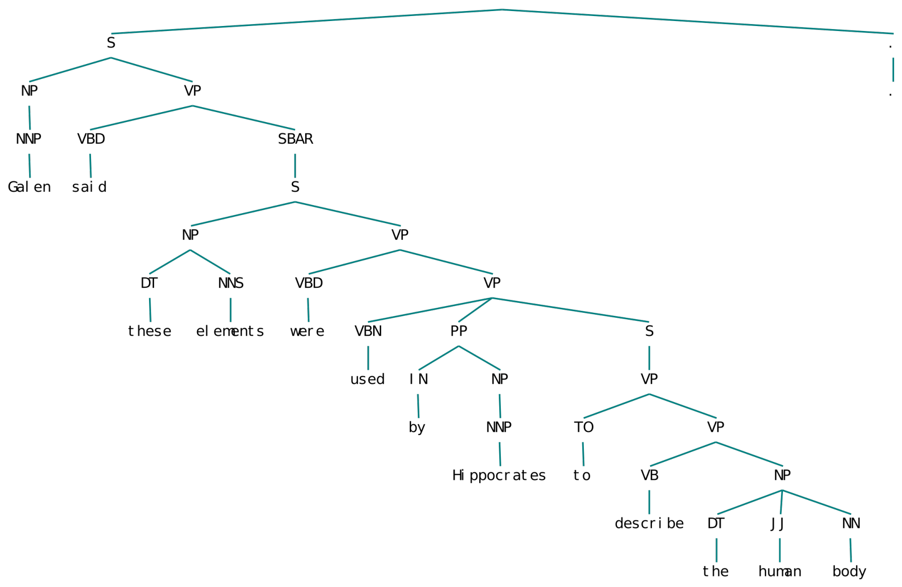

Figure 5: The original complex reported sentence (after hand-curation).

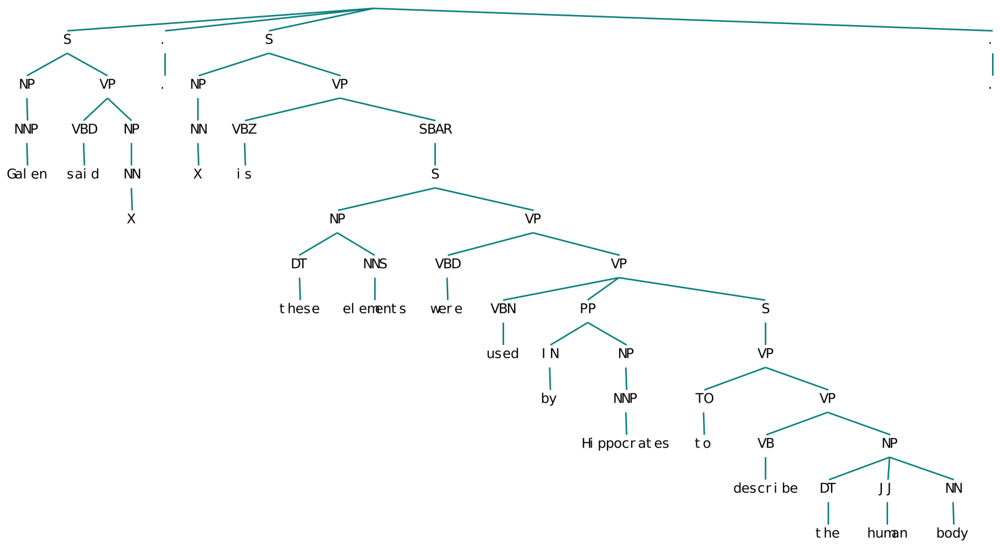

Figure 6: The simplified reported sentence.