APPGATE®

Brainspace

Patented Phrase Detection

WHITEPAPER

Table of Contents

Overview

This whitepaper provides an overview of Brainspace's patented phrase detection. The information below includes a detailed description of Brainspace's unique phrase detection process.

Natural Language Processing (NLP)

For a long time we have known that adding phrases to our processing instead of just processing individual terms would significantly improve the quality of our results.

We started by capturing all n-grams of words where n was in the range 1-4. While this identified lots of phrase candidates, it also had many disadvantages including the following:

- It produced too many candidate phrases which slowed down processing.
- The processing slow down kept us from being able to recognize phrases with more than 4 words in them.
- N-grams produced phrase fragments like "a dog", "the dog", "dog in", "dog on", etc.

At first, we used brute force techniques to avoid phrase fragments, but we still had problems with bad phrases making it through the system, and phrase length limitations.

We knew there were natural language processing (NLP) techniques for identifying phrases that should work much better, including parts of speech (POS) tagging, but the best taggers were at least two orders of magnitude slower than our normal processing.

Finally we found a lightweight POS tagger that used a lexicon to map words to their most likely part of speech and used heuristics to improve the original tagging. After a while of using this, we found ways to improve both the lexicon and the heuristics to improve the tagging quality. We also found ways to rewrite the code to speed up the tagging process.

Once we had mostly accurate POS tags for each word, we created a stack of grammar productions that built up hierarchical phrase representation. Finally we built rules into the system to help determine when to count the entire phrase only, when to count the parts of the phrase only, or when to count both. This allowed us to inject many fewer candidate phrases than N-grams would, while at the same time properly capturing parts of phrases when we needed too.

We also added a tokenizer that ran before the parts of speech tagger. The tokenizer allowed us to identify things like email addresses, URLs, numbers, acronyms, etc. before attempting to use the lexiconbased tagger. This improved our tagging accuracy and increased our tagging speed by avoiding trying to tag items when we already knew how to tag them based on their token type.

Geo-Person Separation

The NLP techniques described so far were a significant step forward in phrase quality, but we still had significant problems, particularly with nouns. A simple POS tagger would lump all nouns together as a single type. A more advanced set of tags, like the Penn Treebank Tag set would identify the following kinds of nouns:

- NN Noun, singular or mass
- NNP Proper Noun, singular
- NNPS Proper Noun, plural
- NNS Noun, plural

Because Brainspace uses a tokenizer to identify some specific kinds of words, we already had the following noun types before making changes for geo-person separation:

- NNA Proper Noun acronym
- NNC Proper Noun company name
- NNE Proper Noun email address
- NNH Proper Noun host name
- NNI Proper Noun initial
- NNPH Proper Noun Honorific
- NNTN Noun, twitter name
- NNTT Noun, twitter tag
- NNU Noun, URL

Even with this richer set of noun tags, we would come across undifferentiated blocks of nouns like “New York City Mayor Michael Bloomberg”. To a human it is easy to see that “New York City” is a place, and “Mayor Michael Bloomberg” is a person. Unfortunately, given the parts of speech we could identify, this phrase was a long list of nouns. At first, we thought that we could just improve the lexicon to separate person names from place names, but we soon found that there are many names that are both person and place names. After scouring the internet for lists of people names, and lists of place names, then comparing to see which names are: person names, place names, or both, we came up with over 440,000 person names, over 21,000 place names, and over 64,000 names that could be a person or a place. Each

of these lists are of individual words that could be person or place names, not full names of a person or place.

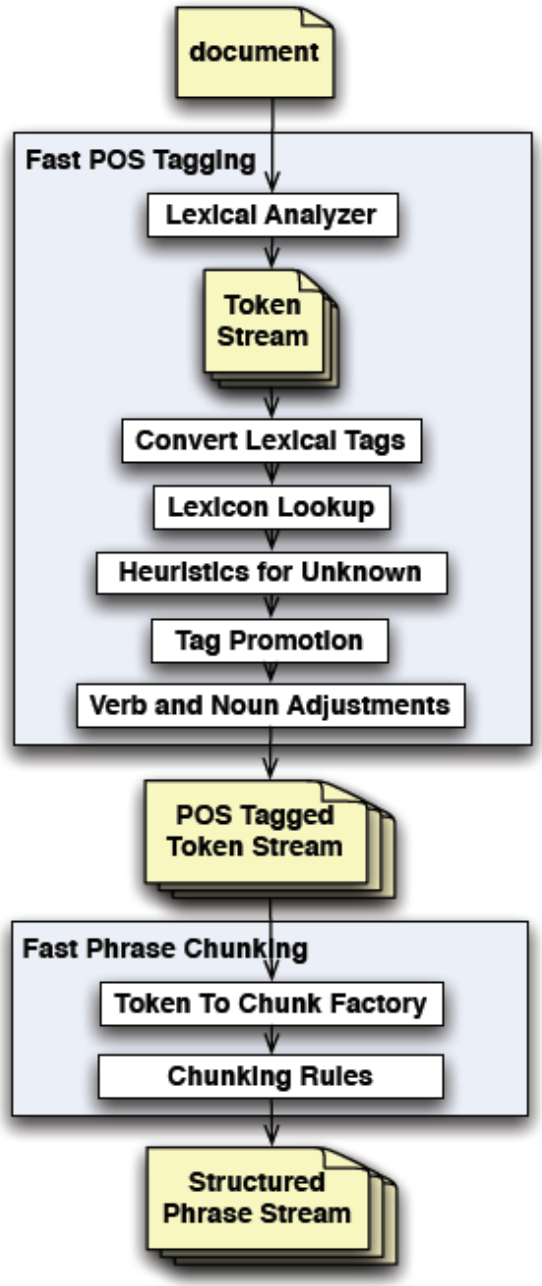
Once we had these lists, we could add the following new noun tags:

- NNPG Proper Noun geo name
- NNPGP Proper Noun geo or person name
- NNPP Proper Noun person name

Adding these tags was not enough. We also had to add special processing. Clumping people names, or place names together is easy. Handling clumps of names that could be a person or place required special processing based on surrounding nouns. If a person or geo name is before or after a clump of terms that could be either, then all the terms are clumped together and take on the more specific type.

The result is that a phrase like "New York City Mayor Michael Bloomberg" is seen by the system as a conjunction of "New York City", "Mayor", and "Michael Bloomberg". These separate phrases, and the super-phrase made by joining the three is much more useful for the kind of analysis performed by Brainspace than a single undifferentiated block, or N-gramsof words taken from that block.

Bloack Diagram 1: Fast NLP with Improved Geo-Person Separation



At the highest level, Fast NLP with Improved Geo-Person Separation has two major steps to it:

1. Fast Parts of Speech (POS) tagging.
2. Fast Phrase Chunking.

These stages are described in more detail in the following sections.

Fast POS Tagging

LEXICAL ANALYZER

- A lexical analyzer that identifies the following kinds of entities:
- Alphanumeric (including common abbreviations, and special words like “c#”, “c++”, and “.net”)
- Alphanumeric that contains apostrophes
- Acronyms
- Initials
- Prefix (e.g. “Mr.”, “Ms.”, “Dr.”, etc.)
- Company (alphanumeric with embedded & like “AT&T” or “P&G”
- Email
- Hostname
- Numeric (numbers optionally including currency symbols)
- Kanji
- Breaking whitespace (families of whitespace that should break phrases: three or more spaces,
- two or more newlines, one tab or form feed, etc.)
- Breaking punctuation
- URL
- Comma
- Other (anything that is not whitespace, and is not one of the other categories)

It is important that the lexical analyzer captures many forms of punctuation that does not break a phrase, such as ‘.’ At the end of prefixes, in emails, in hostnames, in numbers, or in acronyms. It is also important that the lexical analyzer differentiates between whitespace that can be inside a phrase and whitespace

that should break a phrase. Avoiding false breaks in phrases and false phrases that shouldn't exist are both critical to identifying high quality phrases.

CONVERT LEXICAL TAGS

Determine if the Lexical Analyzer tag is sufficient for selecting a part of speech (POS) tag, or if the tag is

still unknown.

LEXICON LOOKUP

If a POS tag has not been determined already, look the token up in a lexicon that maps words to POS tags. First perform a case sensitive lookup, and if that doesn't work, perform a case insensitive lookup to find a POS tag candidate. The lexicon used by Brainspace goes beyond identifying standard POS types such as those found in the Penn Treebank POS Tags. The Brainspace lexicon adds the following tag types:

- INDT - preposition determiner -- equivalent to "of the". Sometimes found as a single word in foreign languages
- NNI - noun initial -- used for tokens that could be initials as part of a person's name
- NNPG - Proper noun geo name
- NNPGP - Proper noun geo or person name
- NNPH - Proper noun honorific
- NNPP - Proper noun person name

The additional types of proper noun tags for geo, person, geo and person, and honorifics are particularly important for separating person and geographic names. Their importance will be seen in a later processing step.

Heuristics for the Unknown

After Lexical Analysis and Lexicon Lookup, the POS tag may still be unknown. If it as, apply a set of 9 heuristics to determine the POS.

- If the token contains a number, tag as "cardinal number".
- Otherwise if the token ends with "ed", tag as "verb past participle".
- Otherwise if the token ends with "ly", tag as "adverb".

- Otherwise if the token ends with “al”, “ic”, “less”, or “like”, tag as “adjective”.
- Otherwise if the token ends with “ing”, tag as “verb, gerund or present participle”.
- Otherwise if the token ends with “the”, tag as “determiner”. (A typical error is no space before “the”.)
- Otherwise if the token is a legal roman numeral, tag as “cardinal number”.
- Otherwise if the first character of the token is a letter, tag as “plural noun” if the token ends in ‘s’, or “noun” if it does not end in ‘s’.
- Otherwise tag as “symbol”.

TAG PROMOTION

Heuristics for making some small tag type changes.

- If tagged as “noun” and first character is upper case or title case, change tag to “proper noun”.
- If tagged as “plural noun” and first character is upper case or title case, change tag to “plural proper noun”.
- If tagged as any of the “noun” types, and the tag for the previous token was “modal” (would, could, etc.), change to “verb”.

VERB & NOUN ADJUSTMENTS

Heuristics for changing some verbs to nouns and some nouns to verbs.

- If the previous token was tagged as a “determiner” and the current token is a VB, VBD, or VBP, make the current token a “noun”.
- If the previous token is “to” and the current token is a “noun” that is not capitalized, make the current token a “verb”.

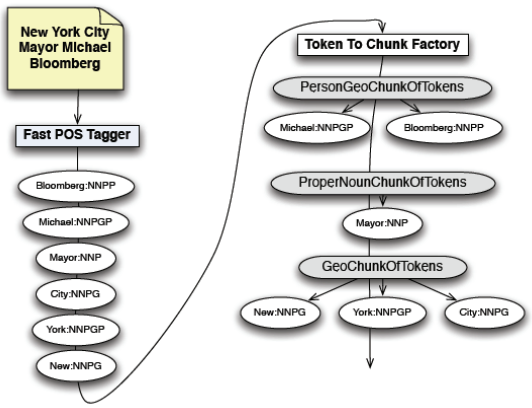
Fast Phrase Chunking

Fast Phrase Chunking is performed by a phrase aware generative grammar using a stack of non-recursive productions, and a factory that provides initial groupings of words prior to applying generative grammar rules. The grammar rules are applied in-order as matches are found. The order the productions are specified in is significant.

SEPARATING NOUNS

The fast phrase chunking works together with the extended noun tags provided by the Fast POS Tagging system to separate people, places, and other nouns from each other more quickly than more standard algorithms allow. For example, the phrase "New York City Mayor Michael Bloomberg" contains people and place nouns. Let's look at how the systems work together to parse this phrase.

Bloack Diagram 2: Fast Phrase Chunking



The diagram above shows the output of tagging the words in the phrase we are parsing, and then running through the first stage of phrase identification.

First the Fast POS Tagger produces a stream of tokens. Each token has a value and a type, with the tokens shown in the diagram as "New:NNPG", "York:NNPGP", "City:NNPG", etc. The types used in the above diagram are as follows:

- NNP - A proper noun
- NNPG - A proper noun for a geographic location
- NNPGP - A proper noun that could be for a geographic location or a person
- NNPP - A proper noun for a person

PARSING STREAMS OF TOKENS WITH PARSING CHUNK FACTORY

The stream of tokens is passed to the Token-To-Chunk-Factory. The factory knows which types of words should be grouped as phrases, and which types should be treated separately. In this case, with all proper nouns, all of the words could be grouped together as a single phrase, but the extended parts of speech types are going to help this system identify useful sub-phrases.

- New:NNPG
- New York City
- Mayor Michael
- Bloomberg
- GeoChunkOfTokens
- York:NNPGP City:NNPG
- Fast POS Tagger
- New:NNPG
- York:NNPGP
- City:NNPG
- Token to Chunk Factory
- Mayor:NNP
- Michael:NNPGP Mayor:NNP
- PersonGeoChunkOfTokens
- Bloomberg:NNPP
- Michael:NNPGP Bloomberg:NNPP
- ProperNounChunkOfTokens

The first token the factory sees is "New:NNPG". This token is a proper noun for a geographic location, and the factory creates a GeoChunkOfTokens for this token. Next, the factory peeks ahead at the streamto see how many other tokens can be part of this GeoChunkOfTokens it just created. "York:NNPGP" is a proper noun token that could be for a person or geographic location and can be added to the GeoChunkOfTokens. "City:NNPG" is another geographic proper name and is added to GeoChunkOfTokens.

The next token, "Mayor:NNP" is not a candidate for a geographic proper name, and so instead of adding this token to the GeoChunkOfTokens, a ProperNounChunkOfTokens is created.

Just like it did for the GeoChunkOfTokens, the factory looks ahead in the stream to see if there are more proper nouns that can be added to the ProperNounChunkOfTokens. The next token "Michael:NNPGP" is a proper noun, but it is more specifically a proper noun that could be naming a specific geographic location or person, and so is not added to the ProperNounChunkOfTokens.

Since "Michael:NNPGP" could be part of a person or place name, a PersonGeoChunkOfTokens is created. At the time this kind of chunk of tokens is created, people proper nouns, place proper nouns, or proper nouns that could be either can be added to it. The next token, "Bloomberg:NNPP", changes the behavior of the PersonGeoChunkOfTokens. Since the Bloomberg token can only be a person proper noun, once it is added to the PersonGeoChunkOfTokens, that particular chunk will only accept additional proper nouns that could be proper nouns for a person.

After the factory is done, we have the following chunks with the following types:

- "New York City":NNPG
- "Mayor":NNP
- "Michael Bloomberg":NNPP

PRODUCTIONS THAT WEAKEN TYPES

Most of the productions in the grammar are binary or trinary productions. A binary production might look like X + Y -> Z, which means if you find something of type "X" in the chunk stream followed by something with type "Y", replace those two chunks with a single chunk of type "Z" that contains "X" and "Y". Similarly a trinary production W + X + Y -> Z means if you find types "W", "X", and "Y" adjacent to each other and in that order in the chunk stream, replace them with a chunk of type "Z" that contains the "W", "X", and "Y".

Some concrete examples follow:

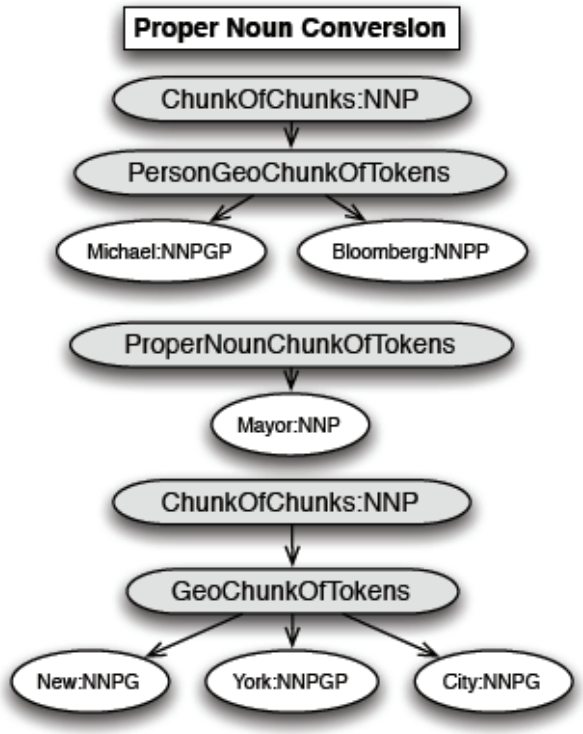
- NNPP + NNI + NNPP -> NNPP (e.g. "John", "Q.", "Smith" can be grouped together as "John Q. Smith" and given type NNPP)
- NNI + NNPP -> NNPP (e.g. "J. K." "Rowling" can be grouped together as "J. K. Rowling" and given type NNPP)

While these kinds of productions are the most common in the grammar, there are other kinds as well. When it comes to dealing with proper nouns, an important kind of production is one that converts one

type to another, usually used to convert from a specific type, to a more general type. For our example, the following two productions are important:

- NNPG -> NNP
- NNPP -> NNP

Bloack Diagram 3: Connecting Specific Nounds to General Nouns



After converting the more specific proper noun chunks in the stream, the chunk stream contains the following:

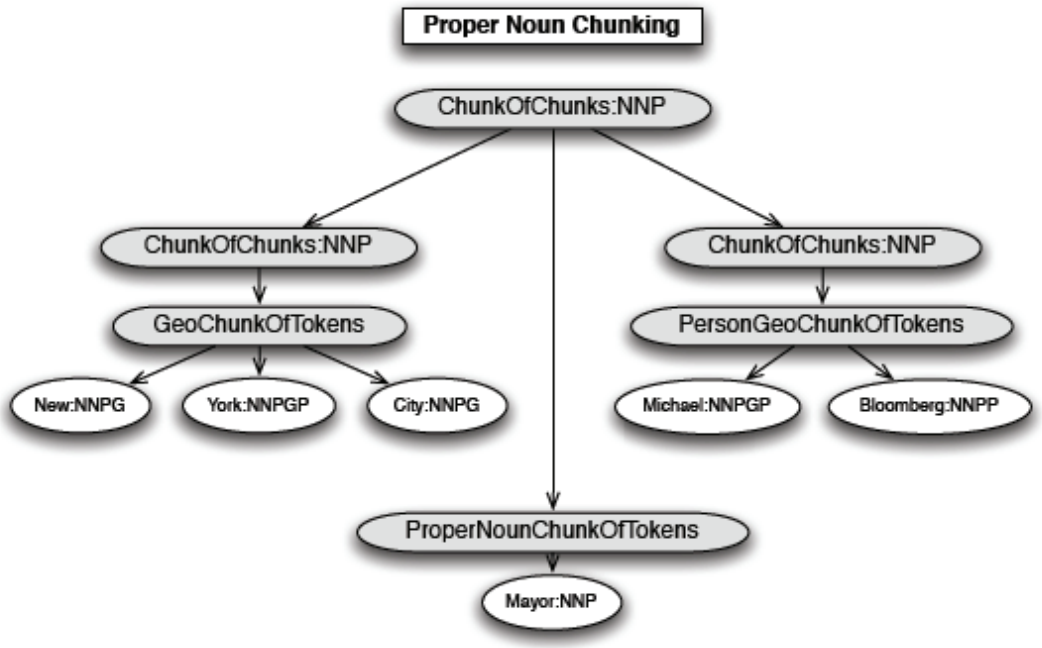
- "New York City":NNP
- "Mayor":NNP
- "Michael Bloomberg":NNP

Now the three identified sub-phrases have the same type.

OPEN ENDED PRODUTION FOR GROUPING PROPER NOUNS

Another production that doesn't fit the binary and trinary production model is one used for proper noun grouping. In this case a production is used that groups as many adjacent proper nouns together as it can.

Bloack Diagram 4: Grouping Proper Nouns



After grouping proper nouns together, our original phrase "New York City Mayor Michael Bloomberg" is reconstructed as a proper noun phrase.

COUNTING CHUNKS

Brainspace does all of this processing identifying phrases and sub-phrases so it can do a better job of identifying and counting the words and phrases in documents. These improved word and phrase counts improve the clustering and LSA results for Brainspace's products.

In our example, "New York City Mayor Michael Bloomberg", standard n-gram word and phrase statistics, where n is 2 or 3, would have produced the following:

- New York City
- York City Mayor
- City Mayor Michael
- Mayor Michael Bloomberg
- New York
- York City
- City Mayor
- Mayor Michael
- Michael Bloomberg

Some of these phrases are meaningful to a human being, and some aren't.

PROPER NOUN CHUNKING

- ChunkOfChunks:NNP
- New:NNPG
- GeoChunkOfTokens
- York:NNPGP City:NNPG
- Mayor:NNP
- ProperNounChunkOfTokens
- ChunkOfChunks:NNP
- PersonGeoChunkOfTokens
- Michael:NNPGP Bloomberg:NNPP
- ChunkOfChunks:NNP

By using Fast POS Tagging and Fast Phrase Chunking, in roughly the same amount of time it would take to produce ngrams, Brainspace produced the following:

- New York City Mayor Michael Bloomberg
- New York City
- Michael Bloomberg

Brainspace produces fewer phrases, and the phrases that are produced are more meaningful to human beings.