## Technology Score

We define a technology as “Something that can be used in scientific research as a resource that helps further the goal of the research”. A technology term is a phrase matching any of the following criteria:

* Artifact – a man-made object produced as the result of a scientific manufacturing process (e.g., *electron microscope*, *computer keyboard, control rods*, *micro-processors*), this includes drugs like *5-fluorouracil*.
* Process/technique – the name of a method or process for creating an artifact or doing technical work (e.g., *duty cycle control*, *electron microscopy, adjuvant chemotherapy*), this includes theories and algorithms (*random matrix theory*, *support vector machines*) and medical technologies (*MIRNA target site*, *lambda DNA replication*, *hairpin probe*).
* Field – the name of a discipline or scientific area relating to the production of artifacts or processing (e.g., *biotechnology*, *construction engineering*)

Examples of non-technologies, on the other hand, are natural kinds (*iron*, *carbon* *dioxide*), measures, and common non-technical artifacts (*chair*).

In the remainder of this document we look at the steps used to extract technologies from a text corpus. It should be noted that technology classification is done corpus wide, that is, a term is classified as a technology if it is used as a technology throughout the corpus. Individual instances of the term are not classified as technologies or non-technologies. As a result of this, the classification makes sense only on fairly homogenous corpora since the intuition is that technical terms tend to be unambiguous on those corpora.

**Pre-processing**

To extract technologies we first tokenize the text, split it into sentence and assign part-of-speech tags. We then extract terms, which are defined as a simple noun phrase composed of an optional adjective followed by one or more nouns. We use a simple specialized noun chunker that looks for patterns of the form (?ADJ NOUN+). A predefined list of about 250 common, non-technical adjectives is used to trim adjectives that are not likely to be used as part of a technical term, this list is available at <https://github.com/techknowledgist/tgist-features> in the *resources* directory in the *en\_jj\_vb.noise* file.

**Training data**

The technology score gives a handle on whether a term in a set of documents is a technology term or not. The technology tagger uses the Maximum Entropy classifier from the Mallet toolkit and a set of several thousand seed terms, both technologies and non-technologies. These seed terms are used to automatically create training instances from any set of data and consequently models for any data set can be created. We opted for a machine learning approach because of the broad scope and the dynamic nature of the patent domain (which was our primary corpus). And Maximum Entropy classifiers in particular have been successfully applied to the similar task of named entity detection for concept classes like people, places, and companies.

For creating the labeled training data, we used a method known as distant supervision. We split a set of 500 randomly sampled patents into a training set of 490 and a test set of 10. For both sets, we extracted all candidate technology phrases, ordered these terms by frequency of occurrence and presented them out of context as a list of terms to the annotator. Two annotators marked all terms with ‘y’, ‘n’, or ‘?’, following published annotation guidelines and making Wikipedia and Google queries if needed to understand the meanings of unknown terms. Initially, we annotated the top 2000 terms. For inter-annotator agreement, we got a Cohen’s Kappa score of 0.52, indicating moderate agreement. A single label was chosen for each annotated term.

**Feature extraction**

We used two classes of linguistic features, intrinsic and extrinsic. Intrinsic features included the first and last words within a phrase, suffixes, and the part of speech signature of the phrase. Extrinsic features were based on the phrase’s immediate lexical environment and document section and included 2 and 3-grams to the left and right of the phrase, as well as the closest preceding verb/verb-particle, noun, and adjective. Features are extracted for each term. Here is a full list of all features used for English:

|  |  |
| --- | --- |
| prev\_n2 | previous two tokens |
| prev\_n3 | previous three tokens |
| next\_n2 | next two tokens |
| next\_n3 | next three tokens |
| next2\_tags | next two part-of-speech tags |
| prev\_J | adjective immediately before the term |
| prev\_Jpr | first adjective-preposition combination to the left of the term, but within four tokens |
| prev\_Npr | first noun-preposition combination to the left of the term, but within four tokens |
| prev\_V | verb or phrasal verb preceding the term |
| prev\_VNP | combination of prev\_\_V and prev\_Npr, finds *increase the speed of* in *increase the speed of the computer* |
| first\_word | first word of the term |
| last\_word | last word of the term |
| tag\_sig | tag signature of the term, a list of part-of-speech tags |
| suffix3 | last three letters of the last word of the term |
| suffix4 | last four letters of the last word of the term |
| suffix5 | last five letters of the last word of the term |
| doc\_loc | location in the document, measured by the sentence offset |
| section\_loc | location in the section, includes the section header and a 0 for first sentence in the section and a 1 for any other sentence in the section |
| sent\_loc | location in the sentence, measured by a range of token offsets |
| plen | number of words in the term |

There are no syntactic features like path\_to\_top because we decided to not run a full parse for processing time considerations.

**Creating the model**

Training instances for the MaxEnt classifier were constructed in three steps.

1. Extract features for all term occurrences in a corpus. The terms used are those that appear in the seed list with positive and negative instances.
2. Create sentence-level instances by combining the term labels with the extrinsic features for each annotated term in a document.
3. Combine the feature sets for all occurrences of the same term in the document into a single document-level training instance. Add the intrinsic features to this instance.

At the end of these steps we have a feature vector for each term in each document. These vectors are input to the regular Mallet process of creating the model.

The process outlined here can be adapted fully automatically for any corpus. All that is needed is the seed list and this list is intended to be general in the sense that a targeted model can be created for any domain.