Classification and prediction of evolving technology

Norman H. Packard[†]*, Nick Gigliotti[‡],
Ananthan Nambiar[‡], Mark A. Bedau[‡]*

[‡]Reed College, [†]ProtoLife, Inc.,

*contact authors: n@protolife.com, mab@reed.edu

Abstract

We study industry classifications that emerge from semantic connections between patent records, as the patent corpus grows from year to year. The semantic connections between patents are obtained using topic modeling, a statistical analysis of text data, from patent titles abstracts of the roughly 5 million patent records we examine, spanning from 1976 to 2014. The statistical analysis extracts semantic information from patent text and uses it to form categories on the basis of semantic similarity of patents. These emergent categories may be used for classification, providing a purely statistical alternative to the US Patent Office's use of pre-defined categories for classification. Categories evolve as new patents are issued, and we track this evolution over time to find a variety of dynamical phenomena: birth, merging, splitting, and dying of categories. Tracking the evolution of patent categories also enables a form of prediction; important terms may be identified, and their importance may be quantified and predicted, yielding a view of the emerging nature of technology in the near future.

Keywords: patents, patent record, patent classification, evolution of technology.

Contents

1 Introduction

Technology is an inherently generative process, and technological innovations are naturally reflected in the patent record since criteria for awarding a patent to an invention include that it be "new and useful" (U.S. Code (1952)). The patent documents encode a patent's meaning; the present work seeks to extract that meaning quantitatively, for a large population of patents spanning 39 years, and to build a classification system built on this quantitative extraction of meaning.

The innovative nature of patented technology makes its study difficult because of the emergence problem: properties of the patent record emerge with the advent of new technology, and cannot be known ahead of time, so a quantitative statistical analysis must be flexible enough to recognize and incorporate new properties and categories as they emerge over time. We address the emergence problem by using *learning algorithms* that are capable of pattern recognition that provides the statistical analysis with categories on which statistics may be computed. Once the categories are discovered, we use them to infer classification of patents, observe how such classifications evolve through time, and study the emergence of semantically novel classes that indicate strongly innovative events.

Patents are classified by the USPTO according to a taxonomy of classification codes; a classification is assigned when each patent is granted. These classification codes are determined by patent examiners, and drawn from a fixed hierarchical classification schema. The classification codes specify a range of detail, from six categories at the coarsest level, to 454 classes, $\sim$16,000 subclasses, and $\sim$150,000 finer grained classes. From this, there have been coarser-grained classifications developed for the National Bureau of Economic Research (Hall et al. (2001)). The classification codes used by the USPTO may (and does) change from time to time, as the advent of technological innovation forces the acknowledgement of new categories.

In the present work we provide an alternative to the USPTO's classification scheme, based on identification of clusters of semantically nearby patents. We view the resulting patent taxonomy as an emergent phenomenon that springs directly from the semantic content of patents.

Our approach starts with a collection of documents, each document being a patent's title along with its abstract, for every patent in the 39 years from 1976-2014 (around five million documents). Category formation is comprised of two steps: (i) embedding all documents in *technology space*, a vector space constructed using statistical text analysis based on topic modeling described below, with an embedding specifically designed to place semantically close documents close to each other in the vector space, and (ii) clustering the points in the vector space, with each cluster being a collection of semantically nearby patents. We identify each cluster as a category; each category has a location in the space defined by the centroid of that category's cluster of patents. In the present work, we form the technology space embedding using all 39 years of data, but we study category formation via clustering in time chunks, to elucidate how patent categories evolve over time. We have chosen to break up the 39 years into temporal chunks containing 50,000 patents in each chunk, which

yields 100 time chunks for the roughly five million patents. In our notation below, time t will be an integer between one and one hundred. Note that the later time chunks will tend to comprise less calendar time than earlier ones, because the number of patents issued per year is generally increasing.

Once categories have been learned, the categories may be queried, e.g. for an out-of-sample test patent, by embedding it with the same learned mapping, and observing which cluster centers it is near. Using this approach, category membership corresponds to a (relatively) short distance to cluster centers, and is hence naturally a continuous quantity. By varying the distance threshold to define category membership, the test patent might be in several categories at once.

Besides providing an embedding of every patent document (title plus abstract), technology space also conveniently provides an embedding of every word from the the patents (i.e. the vocabulary). We can obtain keywords for each cluster by observing the words closest to the cluster centers. These lists of keywords enable us to check whether the statistically defined clusters make human-intuitive sense. The technology space embedding also provides a method for determining what categories are similar from one year to the next, based on their distance in the embedding space.

In the subsequent sections, we will describe the details of the process to obtain categories and keywords, then we examine structure of the categories and their trajectories. Finally, we address the question of whether some aspects of the patent category trajectories might be predictable.

2 Categories from clustering

After the `doc2vec` model has been constructed, and all documents are located in the embedding space, we can now look for structure in this cloud of points. For this purpose we use a clustering algorithm. There are several different types of clustering algorithms; we chose k-means clustering because of speed, for the amount of data. The implementation used here comes from the python *sklearn* package (Pedregosa et al. (2011)). For the illustrations in this section, clusters were obtained using k-means clustering with $k = 25$, in a projection of documents into a 300-dimensional semantic vector space created using `doc2vec`. The cluster centroids were obtained as described below. Each time chunk has 25 centroids over 100 time chunks, yielding 2500 centroids points total.

After clustering is performed for a given time chunk of data, the `doc2vec` embedding enables us to locate each cluster in the embedding space. A cluster contains document vectors $\{d^1, ..., d^n\}$ in the embedding space, each document i having coordinates d^i_j, with j going from 1 to d. We define the cluster's position to be its centroid, $c = (c_1, ..., c_d)$, with the coordinates of c simply given by the mean of the cluster's document coordinates, $c_i = (\sum_j d^j_i)/n$. The k-means clustering algorithm is run on data from each time chunk to form k clusters every time chunk, which we will label $c^{t,i}$.

The embedding of words and documents in a vector space enables us to compute dis-

tances, because the vector space comes with a metric. We choose to use cosine similarity metric on the embedding space, so that the distance between two clusters is $d_{\cos}(c^1, c^2) = 1 - c^1 \cdot c^2/(||c^1||\,||c^2||)$ (note that before computing clusters and measuring distances, we normalize the positions of all documents in the embedding space to lie on the unit sphere, which makes cosine distance a reasonable choice).

To get some intuition of the cluster meaning, and to see if clusters correspond in any way to human notions of categories, we list in Table 1 a few clusters with their 20 nearest words in the embedding space. The words listed in the table have been *stemmed*; an algorithm has extracted a stem of words that might occur in several forms (e.g. singular vs. plural), to sharpen statistical counts of the word. Stemming details are described in Appendix A.

To visualize structure in technology space, we project down to a two dimensional space, using the t-Distributed Stochastic Neighbor Embedding (t-SNE) algorithm (Maaten and Hinton (2008)). The t-SNE projection displayed in Figure 1 shows clear evidence of the existence of connected groups of cluster centroids.

The cluster structure is elucidated by coloring the cluster centroids in different ways. In Figure 1(a), the centroids are colored by time; ranging from white in early years to black in later years. Observing structures with a clear white-to-black sequence within the structure is clear evidence that clusters in subsequent time chunks are nearby in the embedding (to be precise, in the `doc2vec` embedding followed by the t-SNE embedding).

Figure 1(b) shows the results of running a *second* clustering algorithm on the centroid points, in technology space to search explicitly for groups of nearby cluster centroids. For this purpose we used the HDBSCAN clustering algorithm (again from the python *sklearn* package (Pedregosa et al. (2011)). All the centroids in each centroid cluster have the same color, and each centroid cluster color is unique. Many of the macro-structures picked out by the eye are identified and colored as an centroid cluster (for example, the six structures in the right and the bottom). However, other macro-structures picked out by your eye are combinations of a number of space-time centroid clusters (for example, the combination of worms at the center top, the middle, and left of the middle).

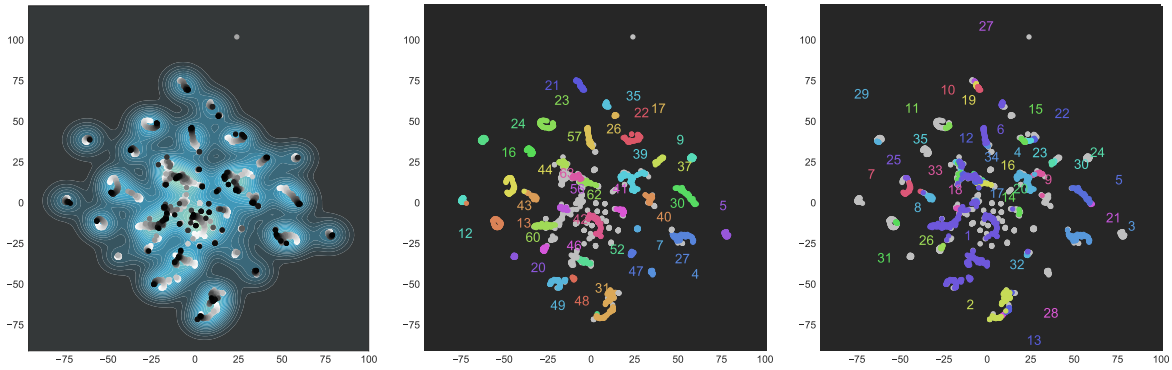
Figure **??**(b) numbers the centroid clusters found in Figure 1(b), and labels them with a human-generated summary provided by the authors, from the `doc2vec` keywords near each centroid (as the examples presented in Table 1) N.B. Keyword lists for many centroid clusters have easily recognizable content, but some keyword lists are more difficult to summarize (e.g., worms 8 and 10).

3 Cluster dynamics

The k-means clustering algorithm works independently, on each time chunk, to identify clusters. It may, however, be the case that in a cluster in one time chunk is close to a cluster in the following time chunk, in which case we may think of the two clusters as being connected in time. We see evidence for this in Figure 1(a). In fact, a cluster may also

Cluster	Worm	human label	Nearest n-grams ($1 \leq n \leq 4$)
$c^{10,8}$	62	conveyances, parts & accessories	tabletop cargo_bed stroller reclin outrigg scooter crutch fifth_wheel ski_bind walker shoulder_strap bogi footrest armrest trailer_hitch jib linkag crib camper seat_cushion
$c^{10,16}$	24	electrochemistry (batteries)	lithium_salt iridium non_aqueou_electrolyt_secondari lithium_batteri selenium pyrrol substitut_aryl lanthanum heteroaryl lithium_ion_batteri non_aqueou_electrolyt sulphid biphenyl dioxid ruthenium schottki_barrier lithium_secondari_batteri electroless
$c^{11,23}$	48	polymers & solvents	polyacryl acryl_methacryl unsatur_monom acryl_copolym methacryl phenoxi methyl_methacryl phosphon_acid mercapto tetrahydrofuran thermoplast_polyurethan acryl_monom meth_acryl fluoroalkyl vinyl_ether methacryl_acid polyalkylen_glycol hydrogen_methyl
$c^{12,24}$	39	computers & communication	dma microinstruct requestor opcod subscript pbx arbit arbitr microcod microprogram argument semaphor decrypt encrypt_decrypt cryptograph spoken prefetch deleg callback dialogu
$c^{13,25}$	22	optics	achromat monochromat beamsplitt meniscu_len linearli_polar incoher aspher pupil confoc blue defocu focal_plane infra_red near_infrar fundu dichroic microlen_arrai acousto_optic circularli_polar red
$c^{14,5}$	31	semiconductors & coatings	epi inp poli silicon_oxynitrid oxynitrid amorph_carbon quaternari hard_mask blue gallium_arsenid microcrystallin epitaxi_grown benzotriazol electroless electrophotograph_photoreceptor selenium sin polycrystallin
$c^{29,20}$	26	images	black_white grai pictori gamma_correct silver_halid_color_photograph binar grayscal foreground halfton jpeg white_balanc monochrom menu_item half_tone chromin subtitl achromat preview radiograph handwritten

Table 1: Cluster keywords: the 20 nearest stemmed words and n-grams for some cluster centroids in some time chunks; e.g., $c^{10,8}$ is the eighth of the 25 clusters from the tenth time chunk.



(a) (b) (c)

Figure 1: (a) Each dot is the centroid of a cluster of patents issued in each of the 100 time chunks that range from 1976 to 2014, projected to two dimensions using t-SNE (see text for description). To reflect time, centroids are colored from white (earlier time chunks) to black (later time chunks). (b) 64 clusters of centroids found with a second application of clustering, indicating structures of nearby centroids in technology space. Each centroid cluster has its own color. (c) Centroid clusters colored by cluster trajectory, as described in the text (also see Fig. 2 for cluster trajectories).

have connections between itself and other clusters in the same time chunk. In this section we explore connections between clusters in time by defining cluster connection in terms of distance between cluster centroids in the 300-d `doc2vec` embedding.

As discussed above, the locations of each centroid for each time chunk, $c^{t,k}$ may be easily computed. We may then compute intra-time-chunk cluster distances $d_{\cos}(c^{t,i}, c^{t,j})$ and inter-time-chunk cluster distances $d_{\cos}(c^{t,i}, c^{t+1,j})$, for $i, j \in \{1, ..., 25\}$ and $t \in \{1, ..., 100\}$. We then say that two clusters are *connected* if they are closer than some threshold distance d_{th}, *i.e.* clusters $c^{t,i}$ and $c^{t+1,j}$ are connected if $d_{\cos}(c^{t,i}, c^{t+1,j}) < d_{th}$, for a specific choice of d_{th}.

Figure 2 shows cluster trajectories that emerge for three different distance thresholds, $d_{th} = 0.02$, $d_{th} = 0.08$, and $d_{th} = 0.12$. In the various cluster trajectories illustrated, we see various detailed phenomena in the dynamics: initiation of new trajectories, termination of trajectories, merging and splitting of trajectories. Trajectory detail is, however, clearly dependent on d_{th}, the distance threshold for defining connections. Small d_{th} (Figure 2(a))means clusters are less connected, trajectories are shorter, and there are more isolated trajectories. Larger d_{th} (Figure 2(b,c)) means clusters are more connected, typically longer, and there are fewer isolated clusters. For typical intermediate values of d_{th} we observe one or two "super-clusters" that include many intra-time-chunk connections as well as inter-time-chunk connections. A detailed picture of a super-cluster with trajectory merging and splitting is shown in the supplementary material.

The construction of cluster trajectories using the distance threshold d_{th} to define cluster

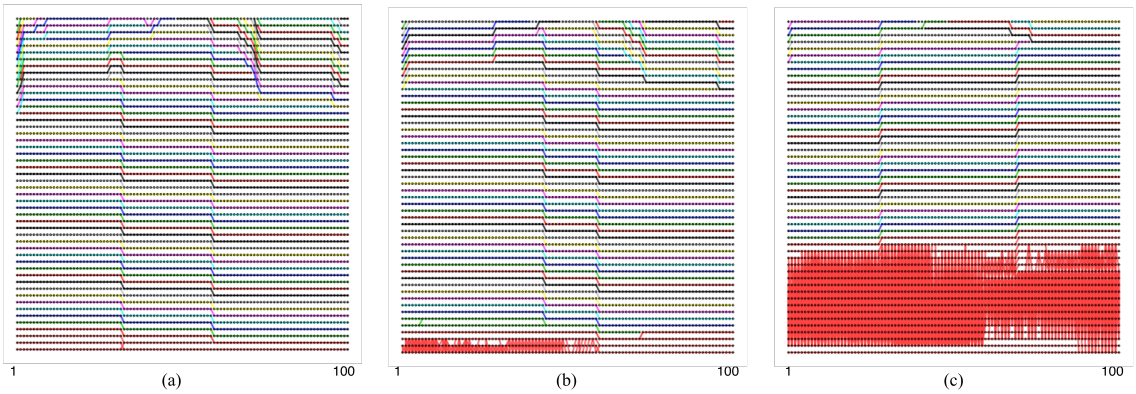


Figure 2: Cluster trajectories: clusters connected in time, for (a) $d_{th} = 0.08$, (b) $d_{th} = 0.12$, and (c) $d_{th} = 0.16$. Time ranges horizontally (in time chunks) from left to right; the 25 clusters found by k-means clustering (with $k = 25$) are arranged vertically in each time chunk. The arbitrary indices of clusters within a given time chunk have been rearranged to minimize tangled trajectories. Clusters are connected if their locations in technology space are nearby enough to fall within a certain distance threshold d_{th}; lower thresholds connect only especially nearby centroids and so create fewer connections, higher thresholds create more connections.

connectedness may be compared explicitly with centroid cluster structure found above, by superimposing the images. Figure 1 (c) shows a t-SNE projection with centroids colored by the identity of the cluster trajectories with $d_{th} = 0.07$ (Figure 2(b)). Short trajectories (those with three or fewer clusters) and isolated clusters are open circles.

Recall that cluster trajectories and centroid clusters are identified by independent mechanisms: cluster trajectories through identification of clusters that are nearer that d_{th}, and centroid clusters by using HDBSCAN clustering on the cluster centroids in the 300-d `doc2vec` embedding. Note: The super-cluster (red in Figure 1 (c), and the bottom trajectory in Figure 2(b)) is the collection of organic chemistry centroid clusters 11, 12, 16, and 17 identified in Figure 1(b). FIX WORM NUMBERS Many centroid clusters have a single color and so correspond to a single trajectory; e.g., centroid cluster 0 (optics), 1 (images), 2 and 3 (integers), 6 (electrical), and 18 (pumps and their parts). Some centroid clusters are colored by connecting the ends of a few threads; e.g., worms 4 (metals), 5 (coatings), and 7 (computers). The two t-SNE regions (center and top) containing many loosely connected small worms consist of many different short threads.

Another way of understanding the meaning of categories obtained from k-means clustering is to compare how they correspond to NBER categories. Figure 3 illustrates the map between k-means clusters and NBER categories. The flow between k-means clusters on the left and and NBER categories on the right is proportional to the number of patents in common. We see that the k-means clusters are not well-aligned with categories, in the

sense that there are very few cases where a k-means cluster maps mostly to a single NBER category.

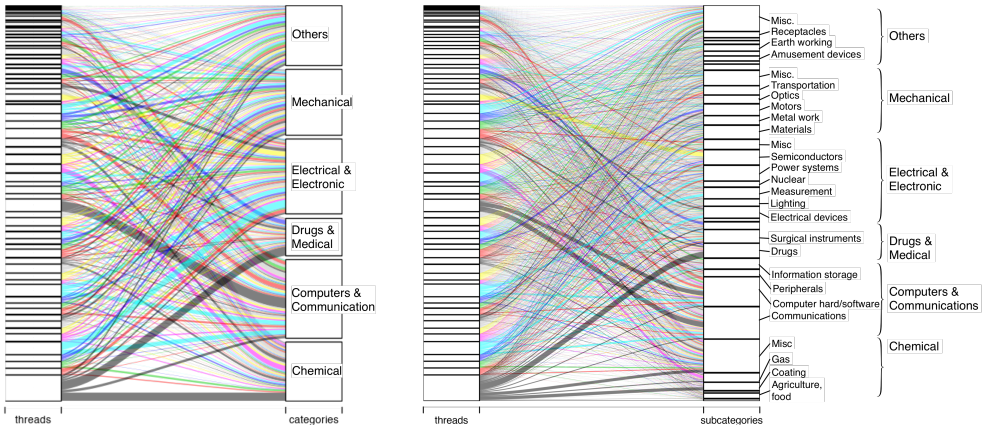


Figure 3: Comparison of NBER categories (a) and subcategories (b) with clusters in the first time chunk, obtained with k-means clustering of patents embedded in technology space. Each figure is a parallel sets plot, where the correspondence between patents in k-means clusters on the left and NBER (sub)categories on the right is shown with a line with a width proportional to the overlap between the cluster and (sub)category, i.e., the number of patents in common.

4 Innovations

Emergence of a new technology class is marked by a characteristic signature in the temporal cluster networks: the start of a new trajectory. Moreover, we can discern from the placement of the new cluster in technology space whether it is close to existing clusters, of far from existing clusters. New trajectories that are far from existing clusters may be regarded as *strong innovations*.

Just as the connection between clusters depends on a distance threshold parameter, d_{th}, the strong innovation criterion depends on a second threshold, $d'_{th} > d_{th}$. In Figure 4, we have chosen to set $d'_{th} = 2d_{th}$. A newly formed cluster centroid is a strong innovation if its $d_{min} > d'_{th}$. We see that there is a continuous stream of new innovations over the 39 year time span, denoted by the triangles (with weak innovations denoted by circles).

5 Materials and methods

In this section we describe the details of how modern statistical text analysis, a field often referred to as "topic modeling", enables us to convert a corpus of documents consisting of

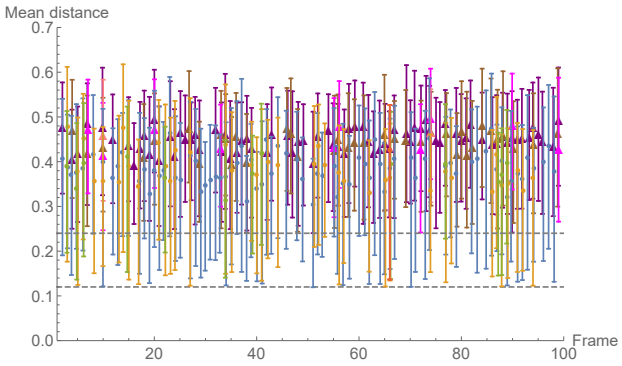


Figure 4: Distance distribution from newly formed cluster centroids to all other centroids, both contemporaneous and past. Shown is the mean distance, with bars for minimum distance and maximum distance. Two thresholds are shown, $d_{th} = 0.12$, which defines cluster connections (as for Fig. 2 (b)) and $d'_{th} = 2d_{th} = 0.24$, which defines a *strong innovation*. A newly formed cluster centroid is a strong innovation if its $d_{min} > d'_{th}$.

words and sentences into a set of points in a high dimensional space, and how we may apply to this embedding clustering algorithms. NB: before applying topic modeling algorithms, we follow standard protocol by preparing the text in a pre-processing step where all words are stemmed. The details are described in appendix A.

Topic modeling technology is now relatively mature, with several off-the-shelf algorithms for statistical textual analysis that aim to extract semantically relevant statistical indicators from a corpus of many documents. The broad aim of all these algorithms is to create a map from raw data consisting of words, sequences of words, and documents comprised of many words, to numerical indicators that quantify semantic structure in the data. For example, if the data set consists of three documents, and two of the documents had many words in common that did not appear in the third document, we might want a numerical indicator that tells us the two documents are semantically nearer to each other than to the third.

We will use a recent algorithm developed specifically for the purpose of mapping documents that are semantically near each other to nearby points in an embedding space: `doc2vec` (Le and Mikolov (2014)), following `word2vec`, a precursor algorithm for embedding words in a vector space (Mikolov et al. (2013b,a)). These algorithms use an artificial neural network to build a model to predict words in the context of a surrounding window of words (thus obtaining word relationship structure unavailable to "bag of word" topic models such as LSA or LDA). This predictive model may then be used to embed documents in a vector space, where the dimensions of the vector space are linguistic features derived from the word-document incidence matrix. `Doc2vec` yields a map of both words and documents into the same embedding space.

We use an open-source python implementation of the `doc2vec` algorithm from the *Gensim* toolset (Rehurek and Sojka (2010)). Several meta-parameters must be specified in

the algorithm's implementation; these are described in Appendix A. One of the most important meta-parameters is the choice of embedding dimension; we have chosen $d = 300$ for all the work in this paper, a value commonly used in the topic modeling literature. Exhaustive exploration of meta-parameters is prohibitive because of the computational resources needed. We have, however, explored meta-parameter variations enough to be convinced that the qualititative structures we identify persist.

6 Conclusions

We have analyzed US patent data from 1976-2014 by using learning algorithms to identify and track emergent structure in the data. We view this approach as essential to address the emergence problem associated with any emergent dynamics. For the patent data, our statistical analysis uses topic modeling tools, combined with statistical clustering, to identify emergent technology classes comprised of patents that are semantically close to each other in a 300-dimensional technology space. We have compared these classes to NBER classes, and find no simple relationship. The classes do, however, seem to be interpretable in human terms that make sense, judging from keywords that are nearby the cluster centroids in technology space.

We find that the clusters in one time chunk can be close to classes in subsequent time chunks, enabling the identification of cluster trajectories. Depending on the distance threshold that defines connections between clusters, we see trajectory formation, trajectory extinction, trajectory merging, and trajectory splitting. We have discovered that in some cases keyword rank within an evolving cluster is predictable, suggesting that we may be able to identify keywords whose importance may increase in the near future.

Appendices

A Implementation and metaparameters

Discussion text preprocessing, stemming, etc. Gensim filters:

```
always do:
to_lowercase
strip_multiple_whitespaces

options:
strip_tags
strip_punctuation
isolate_special_punct
strip_numeric
remove_stopwords
```

strip_short
stem_text

We use a python implementation of the doc2vec algorithm from *Gensim* (Rehurek and Sojka (2010)). There are various hyper-parameters for doc2vec; we take values from a study of them Lau et al. (2014).

doc2vec Hyperparamters:

- dm (0 to use the distributed bag of words training (PV-DBOW) algorithm)
- sg (0 for the Continuous Bag of Word model (for word2vec; irrelevant for doc2vec?))
- hs (0 for no hierarchical softmax)
- negative (e.g = 10 for negative sampling)
- size=300 (the dimensionality of the vectors)
- min count (e.g. 10 $\rightarrow$ ignore words or entity tokens with total frequency lower than 10)
- window (e.g. 10, the maximum distance between the current and predicted word within a sequence)

From Nick:

Here are the default d2v parameters:

```
class gensim.models.doc2vec.Doc2Vec(documents=None,
dm_mean=None, dm=1, dbow_words=0, dm_concat=0, dm_tag_count=1,
docvecs=None, docvecs_mapfile=None, comment=None,
trim_rule=None, **kwargs)
```

And here are the default w2v parameters:

```
class gensim.models.word2vec.Word2Vec(sentences=None, size=100,
alpha=0.025, window=5, min_count=5, max_vocab_size=None,
sample=0.001, seed=1, workers=3, min_alpha=0.0001, sg=0,
hs=0, negative=5, cbow_mean=1, hashfxn=<built-in function hash>,
iter=5, null_word=0, trim_rule=None, sorted_vocab=1,
batch_words=10000, compute_loss=False)
```

References

Hall, B. H., Jaffe, A. B., and Trajtenberg, M. (2001). The nber patent citation data file: Lessons, insights and methodological tools. Technical report, National Bureau of Economic Research.

Lau, J. H., Newman, D., and Baldwin, T. (2014). Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *EACL*, pages 530–539.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In *HLT-NAACL*, volume 13, pages 746–751.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Rehurek, R. and Sojka, P. (2010). Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

U.S. Code (1952). U.s. code 35 101 - inventions patentable.