Document Classification in Biomedical Literature

Celia Siu

2015-08-19

Contents

1 Document Classification

Due to the ever growing number of documents stored in electronic format — such as emails, news stories, research articles, clinical reports, biological databases, and others — proper classification is crucial for information management and knowledge discovery. For instance, with so many biomedical papers being published each month, researchers have a difficult time keeping track of the latest developments or finding important detail previously missed [10]. As a solution, classification implicitly emphasizes the relevant documents such as non-junk emails, news stories entailing particular topics, research papers describing particular fields, and patient charts regarding particular diseases.

In document classification, documents are manually or algorithmically assigned to one or more categories. When done manually, it is time consuming, requires expert labor power, manually implausible, and/or expensive. With an algorithmic approach, [26] calculated — based on a total of 48,194 irrelevant articles and the assumption that pay-per-view access is an average of $30 per article — there is a cost reduction of $1,445,820. [26] also admitted that a human curator is equally able to differentiate between relevant and irrelevant papers based on the title and abstract so the cost is not that high, however the curator is still paid for this task and selecting articles would prevent them from reading the full text and curating the data. In addition to saving in terms of expert labour power and monetary cost, automated methods to classify documents are potentially portable to other domains of knowledge.

In practice, automated document classification has been used for filtering spam emails [29], finding relevant protein interaction articles [36, 30, 6], and performing clinical chart reviews to look for medical reports relevant for study [16].

In this section, we introduced the need for document classification. In section 2 we will describe how documents are represented using features. These features are then used as inputs to classifiers of which will be described in section 3. For this section, we will also go over the advantages and disadvantages of the classifier, follow by how classification performance is evaluated in section 4. In the final section, section 5, we will present research done in document classifications in the field of bioinformatics focusing mainly within the last 10 years.

2 Representation

Text from a document cannot be directly interpreted by a classifier, therefore there is a need to compactly represent the document's content [31]. When choosing the text representation, the choice will depend on what one regards as meaningful units [31]. Typically, terms identified as words are chosen as meaningful units; often this is called the "bag of words approach" to document representation [31]. In this representation, documents occupy a feature space with one dimension per term ("vector space model"), which leads to a feature space with thousands of dimensions and poses a problem for many machine learning algorithms [17].

Other limitations, besides the high dimensionality of the representation, is the loss of correlation between adjacent words and the loss of semantic relationship that exist among the terms in a document [21]. As a solution, one could represent a document by other meaningful units such as:

- **ontology** terms which retains the semantic relationship between terms in a document
- ***n*-grams** consisting of a sequence of n items (e.g. bytes, characters, or words)
- **phrases**
- **RDR** ("Rich Document Representation") which is a logic-based documents representation consisting of a set of logical predicates [20]
- **latent semantic indexing** terms which preserves the most representative features for a document [21]

Furthermore, one could assign term weights for differentiating term importance; these weighing metric include:

- **boolean weighing** which is binary and will give every term appearing in a document equal relevance [27]
- **word frequency weighing** which counts the number of times a term occurs in a document [27]
- **TD-IDF** ("Term Frequency-Inverse Document Frequency") which gives a measure of how frequent a term appears in a document and its interestingness when taking account of how many training documents the term appears in
- **Hadamard representation** which is based on the product of a term in a document and its frequency over all training documents [25]
- **entropy** which assign weights between 0 (if a term appears once in every document) and 1 (if a term appears once in one document) [27]

To address the high dimensionality problem, a subset of relevant features (i.e. those that increase the performance when included in the set of features utilized by a machine learning algorithm [17]) may be selected by feature selection or extracted using feature extraction.

In the following subsections, we will first go over the different steps involved with feature pre-processing; the metrics used for feature selection; and conclude

with how feature extraction differs from feature selection.

2.1 Pre-processing

After having chosen a document representation scheme, the next step is pre-processing for obtaining and reducing the number of features. This involves the following:

- **tokenization** where a document is treated as a string and partitioned into a list of tokens [21]. Example of tokens include words/single terms, n-grams, phrases, RDR, etc.
- **removal of stop words** where frequently occurring, non-informative, and insignificant words such as pronouns, determiners, and prepositions (e.g., 'the', 'that', 'or') is removed [21, 33]
- **stemming** where the root form of the words are obtained by removing the suffixes [33]. Example: "having" will become "hav"
- **lemmatisation** where different inflected or variant forms of a word is grouped to a single term [4]. Example: "having" will become "have"

Depending on the length and number of documents, the number of features obtained from tokenization is large. During pre-processing, non-informative features (i.e. stop words) are removed and redundant terms are treated as a single entity by either stemming or lemmatization.

Removing very rare terms (i.e. terms occurring in less than 3 documents) is also sometimes done during pre-processing [9, 33]. According to [9], this is because very rare terms cannot be used to represent the majority of documents in a corpus.

2.2 Feature selection

To further reduce the number of features — and to overcome the problem of high dimensionality and overfitting of the model to the training data — feature selection is done to obtain a subset of features. Additionally, with a reduced feature set, the system is faster, compact, and comes at a lower computational cost. When the subset of features is well chosen, the classification accuracy is substantially increased and the risk of overfitting is decreased [37].

In feature selection, the original representation of the variables are not altered, but merely a subset of them are selected which offers the advantage of interpretability by domain experts [28]. Consequently, this brings about the problem of finding the right subset of features.

Differing by how the feature space is searched, "filter", "wrapper", and "embedded" are 3 different methods of feature selection. An excellent summary of the advantages, disadvantages, and examples of each search technique is presented in Table 1 of [28]. In brief:

- **filter methods** are independent of the classification algorithm and looks only at the intrinsic properties of the data [28]. In most cases, a feature relevance score is calculated and low-scoring features are removed [28]. Filter methods easily scale to very high-dimensional datasets and are computationally simple and fast [28]. Commonly used filter methods include information gain and chi-squared [17]. Other feature evaluation metrics include the DIA association factor, document frequency, expected cross entropy, gain ratio, Gini index, GSS coefficient, inverse document frequency, mutual information, NGL coefficient, Odds Ratio, Relevancy score, term frequency, and weighted ratio [20, 21, 31]. [28] further mentions feature selection techniques used in the domains of microarray and mass spectrometry.

- **wrapper methods** are dependent on the classification algorithm for they take account of classification accuracy when choosing the subset of features. In this setup, various subsets of features are generated and evaluated by training and testing a specific classification model [28]. This approach therefore tailors subset of features to the specific classification algorithm and is prone to overfitting [28]. Because the number of features is so high, potentially there will be many subsets resulting in greater computational cost associated with wrapper methods [28] and for this reason wrappers are generally not suitable for text classification [20].

- **embedded methods** are a third class of feature selection techniques where the search for an optimal subset of features is built into the classifier construction [28]. These methods offer the advantage of considering interaction with the classification model while being far less computationally intensive than wrapper methods [28].

2.3 Feature extraction

Feature extraction is an alternative method of dimensionality reduction where the original features are mapped into a more effective lower dimensional subspace [24].

A commonly used linear projection method in feature extraction is Principal component analysis (PCA) [24]. Likewise, two other term extraction methods that have been experimented with in document classification is term clustering and latent semantic indexing [31]. In term clustering, words with high degree of pairwise similarity (e.g. synonymous terms) are grouped together and their centroid is used as the representative feature [31]. In latent semantic indexing, singular value decomposition (SVD) is used to compress document vectors (i.e. the collection of features representing a document) into vectors of a lower dimensional space by looking at patterns of term co-occurrences in the original feature space [31].

Overall, artificial terms are created. These synthetic terms may not be intuitively interpretable by domain experts, but they do not suffer the problems of polysemy, homonymy, and synonymy [31].

3 Classifiers

There are different learning algorithms for text classification. In the following subsections we will describe some of these algorithms as well as their advantages and disadvantages.

3.1 Linear Least Square Fit

Linear Least Square Fit (LLSF) is a regression method [31] and mapping approach developed by [41] to estimate the likelihood of association between terms of different vocabularies [41]. In LLSF, each document d_j has two associated vectors [31]:

- an input vector $I(d_j)$ of weighted terms $|T|$, and
- an output vector $O(d_j)$ of weighted categories $|C|$

During training, the category weights are binary [31]. Classification may thus be seen as the task of determining an output vector for a test document given its input vector [31]. Fundamentally, classification boils down to computing [31]:

$$\text{matrix } \hat{M} = |C| \times |T|, \text{ such that } \hat{M}I(d_j) = O(d_j) \tag{1}$$

Usually, matrix $\hat{M}$ — representing the degree of association between category and term — is computed by performing SVD on the training set [41, 31].

Although LLSF is an effective classifier, one of its disadvantages is the higher-than-other-competitors cost in computing $\hat{M}$ [31].

3.2 Rocchio's Algorithm

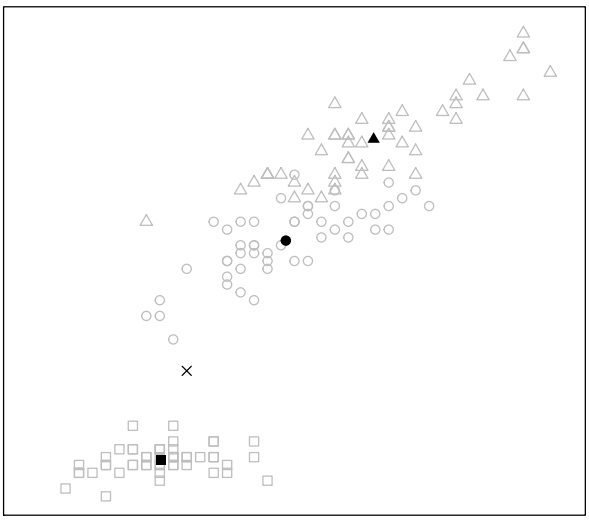

Figure 1: Rocchio's Algorithm for when $\alpha = 1$ and $\beta = 0$ in classifying a new document (x) to one of 3 categories (square, circle, or triangle). In this feature space, the hollow shapes represent training documents and the filled in shapes represent the prototype vectors. In this parameter setting, each prototype vector is equivalent to the centroid of each category. Because the new document is closest to the square prototype vector, the new document will be classified as square.

Rocchio's Algorithm is a vector space method for document routing or filtering in information retrieval [20, 21]. The basic idea of this algorithm is to represent the documents in a given category C_i by a prototype vector (Eq.2) and then classify a new document by assigning the category of the prototype vector to which the test document is most similar to [20, 21].

$$C_i = \alpha * centroid_{C_i} - \beta * centroid_{\overline{C_i}} \tag{2}$$

As an advantage, this method is easy to implement and to understand (i.e. it basically comes down to averaging weights) [31]. The classifier is also fast, efficient, and comes at a low computational cost for only the prototype vectors need to be retained [31]. Because this algorithm is a linear classification method, this method is sometimes too simple and will result in low classification accuracy when the categories tends to occur in disjoint clusters [20, 31].

3.3 *k*-Nearest Neighbors

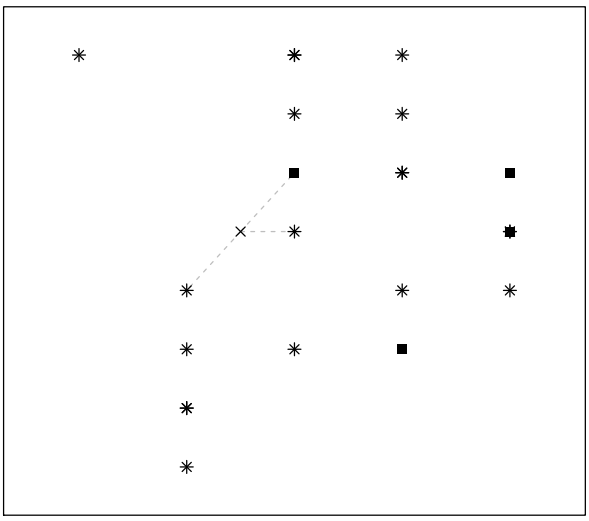

Figure 2: k-NN for $k = 3$ and $distance = Euclidean$ in classifying a new document (x) to one of 2 categories (star or square). In this parameter setting, the closest 3 neighbors from the new document, in this feature space, are 2 stars and 1 square. Because the most frequent neighbor is star, the new document will be classified as star.

k-nearest neighbors (k-NN) is a case-based learning algorithm based on the distance or similarity (e.g. Euclidean distance or Cosine similarity) for pairs of observations [20, 21]. In this method, a test document in the feature space is assigned to a particular category if it is the most frequent category among the k nearest training points [21].

k-NN is effective, non-parametric, and easy to implement [20, 21]. Contrary to Rocchio's algorithm, the feature space is not divided linearly and more local characteristics are considered [21]. However, classification time is long, optimal k and appropriate distance measures are hard to find, and the method uses all features in the distance computation making this algorithm computational intensive when the size of the training set is large [21]. Furthermore, the classification accuracy is degraded in the presence of noisy or irrelevant features [21].

3.4 Naive Bayes

Naive Bayes is a simple probabilistic classifier (Eq.3) that operates under the assumption that features are conditionally independent given the category label (Eq.5) [38]. As an advantage, these simplifying assumptions make the computation more efficient. In its formulation, Naive Bayes calculates the probability that document D belongs to class c_i from prior and class conditional probabilities [21] (Eq.4).

$$P(c_i|D) = \frac{P(c_i)P(D|c_i)}{P(D)} \tag{3}$$

$$P(c_i|D) \propto P(c_i)P(D|c_i) \tag{4}$$

$$P(D|c_i) = \prod_{j=1}^{n} P(d_j|c_i) \tag{5}$$

In addition to being efficient, Naive Bayes is easy to implement and has a low computational cost [20]. Relative to other classification algorithms and as long as the the correct category is more probably than the others, the amount of training data needed to estimate the parameters of Naive Bayes to produce the correct classification is small [20]. On the other hand, due to its conditional independence assumption, the classifier may perform very poorly when the features are highly correlated [20].

3.5 Decision Rule

Decision rules classification method is a bottom-ups approach [31] using rule-based inference to classify documents [20, 21]. In this method, rules — typically in the form of Eq.6 where the `condition` is filled by features of the category and the `conclusion` is the category name or another rule to be tested — are constructed to describe the profile for each category [20]. When the number of features for a category is large, heuristics implementation is recommended to reduce the size of the rule set without affecting the performance of the classification [21]. During classification, every rule in a category's rule set does not necessarily need to be satisfied [20].

```
IF condition THEN conclusion
```
(6)

As an advantage, local dictionaries (able to distinguish the meaning of a particular word for different categories) are constructed for each category during the feature extraction phrase [20]. Decision rules also tend to be more compact than decision tree learners [31]. On the other hand, the drawbacks to the decision rule method include its impossibility to assign a document to a category exclusively, the need for extensive involvement of human experts to construct and update the rule set, and its lack of performance when the number of distinguishing features in a category is large [20, 8].

3.6 Decsion Tree

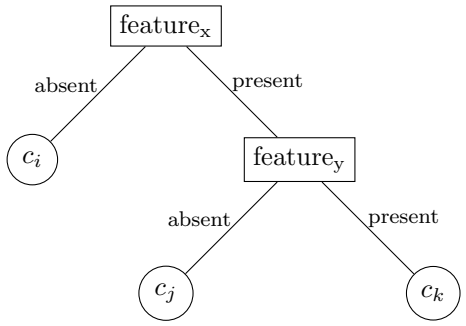

Figure 3: Decision tree for classifying a new document to one of 3 categories (c_i, c_j, or c_k) based on 2 features (x and y). If a new document contains both features x and y, then the category will be c_k; if x is present and y is absent, then the category will be c_j; and if x is absent (regardless of the presence of y), then the category will be c_i.

Decision tree is a widely used inductive learning methods [24] based on an acyclic directed graph with hierarchical structure [8]. It is a top-down method which recursively constructs a decision tree classifier [24] until a terminal node (i.e. leaf) representing a category is reached. At each level of the tree, the algorithm selects a feature (i.e. term) for splitting based on a feature selection metric. The choice of the splitting term is key [31] and examples include the term which has the highest information gain, lowest entropy, lowest Gini index (which is an impurity measure), or lowest classification error. Branching from these internal nodes are labeled by tests on the weight that the term has in the test document [31]. Well known decision tree algorithm include ID3 and its successor C4.5 and C5 [24].

The main advantage of this approach is that it is — even for non-expert users — easy to understand, interpret, and replicate [20]. They are also robust to noisy data [24]. Alternatively, the main disadvantage is that a fully grown tree may be specific to the training data and be prone to overfitting [31, 20]; however, there are methods to reduce overfitting by pruning the decision tree [31].

3.7 SVM

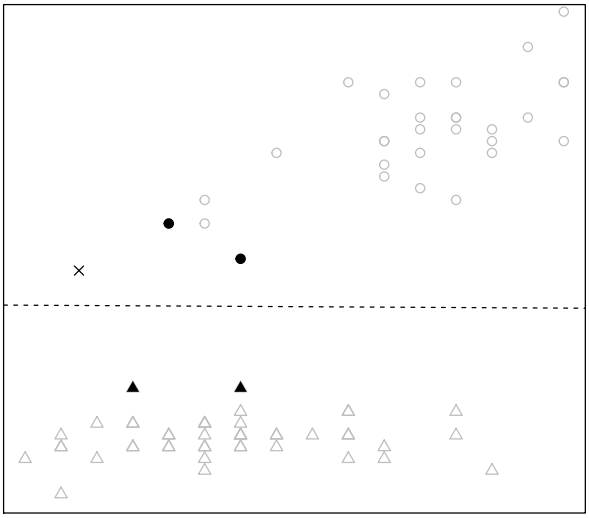

Figure 4: SVM with a linear type kernel in classifying a new document (x) to one of 2 categories (circle or triangle). In this feature space, the circles and triangles represent the training data, the filled in shapes represent the support vectors, and the dotted line represents the decision boundary. Because the new document is on the circle's side of the decision boundary, the new document will be classified as circle.

Support vector machines (SVM) are one of the discriminative classification methods which are commonly recognized to be more accurate [20, 8]. Unlike other classification methods, SVM needs both a positive and negative training set to determine the decision surface (also known as the "hyperplane") that best separates the positive from the negative data in the n-dimensional space [20, 21]. Document representatives closes to the decision surface are called "support vectors" [20, 21] and the performance of the SVM classification remains unchanged if non-support vector documents are removed from the training set [20, 8]. Generally, SVM classification is binary and multiple machines must be trained to detect multiple categories [8].

As an advantage, SVMs can handle documents with high-dimensional input space [20] and eliminates most of the least important features [8] making term selection often unnecessary as SVMs tend to be fairly robust to overfitting [31]. As a disadvantage, the training and categorizing algorithms of SVM is relatively complex and results to high time and memory consumption [20]. It is also important to choose a suitable term weighing function to construct the feature vector because the models are sensitive to the data scale [36].

3.8 Neural Network

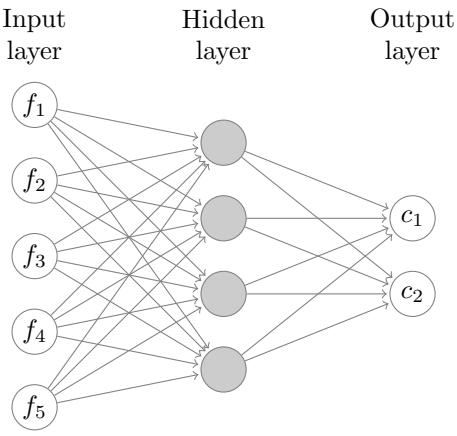

Figure 5: A multi-layer perceptron with 1 hidden layer. This ANN consists of 5 input units, 4 hidden units, and 2 output units. Features are fed into the input layer and categories are returned out of the output layer. The edges between units represent the dependence between units.

An artificial neural network (ANN), or neural network for short, is a heavily connected network of units (i.e. neurons) arranged in layers: input, hidden, and output. The input units of the input layer usually represent the terms and the output units of the output layer represents the categories [21, 31]. The connections between units are weighted to express the strength of connections and dependence relations between particular units [8, 31]. For classifying a test document, the document terms are weighted and assigned to the input units [21, 31]; activation of the units is then propagated forward (Eq.7) through the network to the output layer [21].

$$y = f(\sum_{i=1}^{N} w_i * x_i) \tag{7}$$

where neuron y is in the next layer and is equal to the sum of neurons $x_1 \dots x_N$ of the previous layer. The amount of contribution to y from x_i depends on the strength w_i of its connection.

The simplest type of ANN classifier is the "single-layer perceptron" [31, 20], which contains only an input and output layer thus feeding the input weights directly to the output [20]. The more sophisticated version is the "multi-layer perceptron" consisting of an input layer, one or more hidden layers, and an output layer [20].

The main advantage in document classification with ANN is its ability to handle documents with high-dimensional features and its robustness to correctly

classify documents in the presence of noisy and contradictory data [20]. The drawback of the ANN is their high computing cost (consuming high CPU and physical memory usage) and its extreme difficultly for the average user to understand [20].

3.9 Classifier committees

Classifier committees — also known as "ensembles" — are made up of a group of k individual classifiers and is based on the idea that given a task that requires expert knowledge to be performed, k expert opinions may be better than one if their individual judgements are appropriately combined [31, 21]. The classifier committee is therefore characterized by the choice of its k classifiers and the choice of its combination function [31]. Choices of combination functions include [31, 21]:

- **majority voting** which is the simplest rule where classification is based on the category with the most votes
- **weighted linear combination** where each category is the weighted sum of the judgements from the k classifiers for that category and the weights are reflective of the classifier's effectiveness on the validation examples
- **dynamic classifier selection** where the classifier that performs the best on validation examples is selected and adopted by the committee
- **adaptive classifier combination** where each category is the weighted sum of the judgements from the k classifiers for that category and the weights are reflective of the classifier's effectiveness on the l validation examples most similar to the test document

A special case of the classifier committees is the "boosting method" where the committee is obtained by the same learning method (here called "weak learners" or classifiers which are only slightly correlated with the true classification) [31]. The intuition here is that the k classifiers should be trained sequentially to take into account the weakness of the previous weak learner (i.e. data is reweighed to add more weight — and emphasis — to examples that were previously misclassified) [31]. After all the classifiers have been built, a weighted linear combination rule is applied to yield the final classifier [31].

Overall, classifier committees are easy to implement and understand, but it takes a long time for the results to be returned [21]. They also profit from the complementary strength of the k individual experts, but training these k experts is computationally expensive [31].

4 Evaluation

Evaluation of document classifiers is typically conducted experimentally, rather than analytically due to its subjective character and inherent unformalizability [31]. Some common evaluation metrics include precision, recall, and F_1-measure. After an effective measure is chosen, the classifier can then be tuned by k-fold cross-validation on a train-and-test set.

4.1 Effectiveness metrics: precision and recall

Two measures of effectiveness — i.e. the ability to take the right classification decision [31] — are are precision and recall.

Precision (also known as the "positive predictive value"; Eq.8) in text classification is the probability that, if a random document is classified under c_i, this decision is correct [31]. Precision is also viewed as the "degree of soundness".

$$\text{Precision, } \pi = \frac{TP}{TP + FP} \tag{8}$$

Recall (also known as "sensitivity" or the "true positive rate"; Eq.9) in text classification is the probability that if a random document ought to be classified under c_i, this decision is taken [31]. Recall is also viewed as the "degree of completeness".

$$\text{Recall, } \rho = \frac{TP}{TP + FN} \tag{9}$$

Mirroring sensitivity is specificity
(also known as the "true negative rate")
calculated by $\frac{TN}{TN+FN}$

When considering precision and recall for all categories, 2 methods may be adopted [31]:

- **micro-averaging** where category-specific true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values are first summed and then precision and recall is calculated
- **macro-averaging** where precision and recall are first evaluate locally for each category, and then globally averaged over the results of the different categories

4.2 Combined Effectiveness measures

Evaluating a classifier by precision or recall in isolation does not make sense [31]. Measures which combines precision and recall include:

- **Eleven-point averaging precision** where the classification threshold is repeatedly tuned so as to allow recall to take values of $0, 0.1, \ldots, 0.9, 1$ [31]. The corresponding precision (of the 11 recall values) are then computed and the average is taken [31].
- **Breakeven point** is the value at which precision and recall equal [31]. This value is obtained from the plot of precision as a function of recall when the classification threshold is varied [31].
- F_β**-Measure** for when β — representing the relative degree of importance attributed to precision and recall — is greater than or equal to 0 (Eq.10) [31].

$$F_\beta = \frac{(\beta^2 + 1)\pi\rho}{\beta^2\pi + \rho} \tag{10}$$

  When $\beta = 0$, F_0 is precision and when $\beta = +\infty$, $F_{+\infty}$ is recall. Usually β takes on the value 1 (Eq.11), attributing to the equal importance — harmonic average [36]— of precision and recall. Studies using F_1 include [25, 10, 15, 36, 40, 13, 42, 39, 33].

$$F_1 = \frac{2\pi\rho}{\pi + \rho} \tag{11}$$

Another evaluation measure of classifier performance is combining sensitivity with specificity. For instance, plotting sensitivity (a.k.a. recall or true positive rate) as a function of 1-specificity (a.k.a. false positive rate) for different possible cut-points of a binary classifier [43] resulting to a receiver operating characteristic curve (ROC) is also considered in document classification [18, 38, 36, 37, 32, 26]. The area under the ROC curve (AUC) measures the probability of a threshold classifier that it rates a randomly chosen positive sample higher than a randomly chosen negative sample making it more suitable for applications that require ranking [36]. Unlike AUC, which measures a classifier's performance independent of a cutoff threshold, F-Measure measures a classifier's best classification performance on a specific threshold making it more important for filtering [36]. Furthermore, according to [13], ROC plots are well suited for scenarios of unbalanced classes and show how a classifier performs when tuned for either conservative (high precision) or liberal (high recall) prediction.

4.3 Alternative metrics

Other measures of effectiveness are accuracy and error. For a classification task where the class distribution is uneven (i.e classifying for relevant articles in information retrieval), accuracy (Eq.12) and error (Eq.13) metrics are not

widely used nor recommended because the denominator is large and a trivial rejector (one which classifies every document as irrelevant) would have excellent accuracy [31].

$$\text{Accuracy, } A = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

$$\text{Error, } E = \frac{FP + FN}{TP + TN + FP + FN} = 1 - A \tag{13}$$

Matthews correlation coefficient (MCC; Eq.14), also known as the phi coefficient, is regarded as a balanced evaluation measure for binary classification problems with an imbalanced class distribution [30]. A MCC value of 0 represents an average random prediction and a MCC of 1 represents a perfect prediction [30]. MCC has been used as an evaluation metric in [9, 44, 26, 33].

$$MCCscore = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + TN)}} \tag{14}$$

Alternative measures of effectiveness include measures of efficiency and utility. Efficiency (e.g. training efficiency or classification efficiency) have been seldom used in document classification because of the volatility of the parameters on which the evaluation rests [31]. However, efficiency is useful for choosing among classifiers with the same effectiveness. On the other hand, utility is based on measuring the cost (e.g. gain or loss) of a TP, FP, TN, and FN [31]. For instance, utility is useful for spam filtering where failing to discard a piece of junk mail (FP) is a less serious mistake than discarding a legitimate message (FN) [31].

Other effective measures as listed by [31] include adjacent score, coverage, one-error, Pearson product-moment correlation, recall at n, top candidate, and top n.

4.4 Benchmark collection

Once an effectiveness measure is chosen, a classifier can then be tuned experimentally or evaluated against a benchmark collection [31]. Usually, tuning is performed by k-fold cross-validation. In k-fold cross-validation, the data set is first split into k disjoint sets or folds [13]. The classifier is then trained on $k-1$ folds and tested on the remaining held out fold [13]. This process is iterated so that testing is carried out on each fold exactly once [13].

In the evaluation of a classifier, classification is done against a standard benchmark collection where the target document categories are known. Standard collections include Reuters, 20 Newsgroup, and OHSUMED (Table 1). Other resources include datasets provided by document classification challenges such as the BioCreative, I2B2 Obseity, or TREC challenges (Table 2).

Table 1: Standard collections for document classification for training and evaluation

Collection	Description
Reuters-21578	Collection of 21,578 documents (5 categories) that appeared on Reuters newswire in 1987 [22].
RCV1	The Reuters Corpus Volume I is an archive of over 800,000 manually categorized newswire stories made available by Reuters, Ltd. for research purposes. [23]
20 Newsgroups	Collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups [1].
OHSUMED	Collection of 348,566 MEDLINE references (titles or titles-plus-abstracts) with MeSH terms from medical journals covering 1987 to 1991 [19].

Table 2: Document classification challenges

Challenge	Description
BioCreative II	Participants were asked to classify/rank articles relevant to protein interaction during the first subtask (denoted as IAS) of the PPI portion of the challenge [2]. [36, 40, 6] have made use of this challenge's corpus.
BioCreative III	Similar to its predecessor, participants were asked to classify/rank articles relevant to protein iteration in the first subtask (denoted as ACT-BC-III) of the PPI portion of the challenge [3]. Additionally, in the second subtask (denoted as IMT-BCIII) of the same portion, participants were asked to provide experimental method classification of the articles [3]. [30, 39] have made use of this challenge's corpus.
I2B2 2008 Obesity	As described and used by [17], the goal of this multi-label, multi-class classification challenge is to classify discharge summaries from overweight or diabetic patients as asserting the presence or absence of obesity and 15 related diseases.
TREC Genomics	The purpose of this track was to study retrieval tasks in the genomics domain including gene sequences and supporting documents such as papers and lab reports [5]. [34, 36, 35] have made use of this challenge's corpus.

5 Applications in Bioinformatics

5.1 Feature engineering for improved performance in biomedical document classification

In text classification, features are typically the words and/or concepts present in a text corpus [17]. Potentially, the number of features, even in a small corpus, may be tens of thousands thus necessitating feature engineering — the selection of a subset of informative features and/or the combination of distinct features into new features — for a given classification task [17].

Biomedical terminology is rich and highly complex; and because of this, [18] argues that general-purpose stemming algorithms (e.g. Porter stemmer) are often conservative and could remove informative stems. As an alternative to stemming, [18] proposed an algorithm that omits stemming and instead use the most discriminative (as determined by information gain) substrings (defined to be the consecutive list of symbols within a word) as attributes to classification. Similarly, [37] proposed to directly utilize length-fixed strings (i.e. n-grams) as features to exploit most of the informative segments. In this setup, a document is treated as a string and tokenization is not by whitespace but by the length of a string [37]. In addition, as a substitute from the traditional TF-IDF schema, [37] also proposed a novel feature weighing method by the production of Term Frequency by Maximum Likelihood (TF-ML).

Domain knowledge if often used to guide the feature engineering process [17]. In document classification, features engineered using domain knowledge and semantic relation could be introduced through the use of controlled vocabularies, ontologies, and taxonomical structures such as terms in Medical Subject Headings (MeSH) [40, 30] Unified Medical Language System (UMLS) [17], Gene Ontology [40, 15], or PSI-MI ontology [39]. In addition, domain specific features such as protein assay terms [6] or epitope related expressions [38] could be used. In [6], protein assay terms — as obtained from the "Protein-protein Interaction Assays" article on Wikipedia — were used to identify PPI related documents. In [38], novel rules — that specifically try to capture immune epitope related expressions and group them together — were introduced: peptide sequence group, position ranges stretching less 50 amino acids group, position ranges stretching more than 50 amino acids group, MHC alleles group, and X-mer group.

Finally, another source from which features can be derived from are figures. Figures are often content rich and concisely summarize the most important results or methods used and described in an article [34]. The first exploration of the possibility of using image data in support of document categorization in the biomedical domain, was presented in [34]. These image features were obtained by figure extraction, figure segmentation, subfigure classification into one of four coarsely-defined classes, subfigure clustering into finer groups, and document representation as an image-based feature vector [34].

Overall, the idea here is that better features results to better performance.

5.2 Document classification using clinical text

Clinical data stored in electronic medical records are important to clinical-decision support, comparative effectiveness research, and epidemiological and clinical research studies [16]. One of the keys to accessing clinical data locked in unstructured text is automated document classification [16].

In [17], discharge summaries from overweight or diabetic patients from the I2B2 2008 challenge dataset were classified as asserting the presence or absence of obesity and 15 related diseases, including Hypertension, Coronary Artery Disease (CAD), Congestive Heart Failure (CHF), and Hypercholesterolemia. This was done using an SVM trained on features engineered from the Unified Medical Language System (UMLS) which leverages domain knowledge encoded in the taxonomical structure [17].

In [16], radiology chart reviews were facilitated to screen for and confirm cases of hepatic decompensation for use — i.e. information extraction from the subset of medical records — in a Veteran Aging Cohort study. The system that was trained and tested include the use of decision trees, random forests, ensembles of decision trees, and SVMs on features annotated from syntactic constructs, named entities and their negation context in clinical text [16].

Finally, in [11], the similarity of surface linguistic features (i.e. n-grams) of epilepsy progress notes — from generalized, partial, or unclassified epilepsy — across 3 different hospitals were assess using a SVM. Training on the epilepsy progress notes from one or two hospitals and evaluating on the notes of the third, established that there is a certain degree of uniformity of epilepsy vocabulary across the different hospitals [11]. Furthermore, according to [11], this suggests that a limited number of annotated epilepsy progress notes from each hospital might be enough for developing automated extraction of epilepsy quality measures from clinical narratives.

5.3 Document classification for curation support

Manual literature curation is expensive, labour intensive, and time-consuming [40, 26]. Furthermore, it has become increasingly difficult for curators to keep up with the increasing scientific output [26]. Therefore there is a need for automated systems to prioritize relevant documents for curation.

Immune Epitope Database (IEDB) and ChEMBL are examples of 2 databases who have took advantage of document classification to triage relevant articles for curation. Previously in IEDB, retrieval of relevant epitope-related articles and curation was performed manually by two experienced immunologists [38]. With documentation classification by Naive Bayes [38] — and later by SVM with appropriate parameters for selecting articles in high IEDB categories [32] — the abstract selection process of IEDB reference curation was successfully sped up and its efficiency increased.

Similarly, [26] was able to successfully distinguish between publications, from the medical chemistry literature, that are related to small molecule drug discovery (i.e. compound structures, biological targets, bioactivity data and calculated molecular properties of drugs and drug-like molecules) for the ChEMBL database using Naive Bayes and random forest classifiers. In addition, [26] has also used their method to look for malaria medicinal chemistry publications.

Finding relevant documents using automated document classification does not need to exclusively be for the curation of databases. Other research which does something similar but for a different domain includes [13], who classified abstracts of published literature into categories relevant to peptide research (i.e. relevance in cancer, angiogenesis, molecular imaging, etc) using ensembles of decision trees; [40], who tested Naive Bayes and SVM to recognize context related to transcription factors; [36, 30, 39, 6], who identified PPI-related information in primary literature (respectively using SVM; heuristics and Naive Bayes; logistic regression and SVM; and k-NN); [43], who developed and evaluated a SVM system to identify MEDLINE abstracts referring to pathogen-host PPIs; and [42], who looked for articles mentioning mutations that affect protein stability or function with Naive Bayes, SVM, and Rocchio's Algorithm.

5.4 Document classification in protein research

Knowing the location of proteins within the cell is an important step toward understanding their function and role in biological processes [9, 33], facilitating their purification [15], and potentially in the design of new drugs [44]. Accurate but slow and labour-intensive experimental methods — based on green fluorescent proteins or immunolocalization — can be used identify the location of only readily expressed proteins [9]. Given the large and growing number of proteins of which little is known and many may not be expressed under regular conditions [9], computational methods using document classification and homology have been developed to infer protein subcellular location.

In [9, 15, 33], SVMs were used to predict protein subcellular location using biomedical literature. In this setup, categories represent subcellular locations and articles, that were used for training, were retrieved from protein article links in SwissProt/UniProtKB [9, 15, 33]. In [9], different feature selection methods were compared in-depth for this classification task. In [15], protein subcellular prediction accuracy was improved by removing ambiguous abstracts, unifying synonyms from the Gene Ontology (GO), and leveraging the GO hierarchy to generalize terms. In [33], protein function as based on GO categories was also predicted in addition to protein subcellular location.

Alternative to using biomedical literature to predict protein subcellular location, [44] used just the protein sequence information. In this approach, a protein sequence is treated as a document and features are obtained by dividing the sequence into short overlapping k-mer sequence fragments (i.e. n-grams) which can be mapped to word features in document classification [44]. Training is then accomplished using protein sequences of known subcellular locations. In [44], boosting association rules were used to classify a proteins subcellular location in plants and animals. Similarly, [12] used this setup to developed Allerdictor, a sequence-based allergen prediction tool that models protein sequences as text documents and uses a SVM in text classification for allergen prediction.

6 Conclusion

In conclusion, we have provided an overview of document classification. We described the common methods of document representation following feature selection and feature extraction. We presented 9 machine learning classification methods describing how they work, plus their advantages and disadvantages. We went over classification performance metrics and provided a list of benchmarking collections from corpuses and challenges designed for document classification. In the last section, we concluded with some applications of document classification — involving feature engineering, clinical text, curation support, and protein research — in biomedical literature.

Acknowledgements

Figures 1, 2, and 4 were generated from the iris dataset [7] and the neural network of Figure 5 was generated based on code from [14].

References

[1] 20 newsgroups. Available from http://qwone.com/~jason/20Newsgroups/.

[2] Biocreative ii. Available from http://biocreative.sourceforge.net/biocreative_2_ppi.html#description. Accessed: 2015-08-18.

[3] Biocreative iii. Available from http://www.biocreative.org/tasks/biocreative-iii/ppi/. Accessed: 2015-08-18.

[4] *Collins English Dictionary*. Entry for "lemmatise".

[5] Text retrieval conference (trec). Available from http://trec.nist.gov/tracks.html. Accessed: 2015-08-18.

[6] Ambert, K. H., and Cohen, A. M. k-information gain scaled nearest neighbors: A novel approach to classifying protein-protein interaction-related documents. *IEEE/ACM Transactions on Computational Biology and Bioinformatics 9*, 1 (2012), 305–310.

[7] Anderson, E. The irises of the gaspe peninsula. *Bulletin of the American Iris Society 59* (1935), 2–5.

[8] Bilski, A. A Review of Artificial Intelligence Algorithms in Document Classification. *International Journal of Electronics and Telecommunications 57*, 3 (2011), 263–270.

[9] Brady, S., and Shatkay, H. EpiLoc: a (working) text-based system for predicting protein subcellular location. *Pacific Symposium on Biocomputing 13* (2008), 604–615.

[10] Chen, D., Müller, H.-M., and Sternberg, P. W. Automatic document classification of biological literature. *BMC bioinformatics 7* (2006), 370.

[11] Connolly, B., Matykiewicz, P., Bretonnel Cohen, K., Standridge, S. M., Glauser, T. a., Dlugos, D. J., Koh, S., Tham, E., and Pestian, J. Assessing the similarity of surface linguistic features related to epilepsy across pediatric hospitals. *Journal of the American Medical Informatics Association 21* (2014), 866–870.

[12] Dang, H. X., and Lawrence, C. B. Allerdictor: fast allergen prediction using text classification techniques. *Bioinformatics 30*, 8 (2014), 1120–1128.

[13] Duchrow, T., Shtatland, T., Guettler, D., Pivovarov, M., Kramer, S., and Weissleder, R. Enhancing navigation in biomedical databases by community voting and database-driven text classification. *BMC bioinformatics 10* (2009), 317.

[14] Fauske, K. M. Texample.net, example: Neural network. Available from http://www.texample.net/tikz/examples/neural-network/. Accessed: 2015-08-18.

[15] Fyshe, A., Liu, Y., Szafron, D., Greiner, R., and Lu, P. Improving subcellular localization prediction using text classification and the gene ontology. *Bioinformatics 24*, 21 (2008), 2512–2517.

[16] Garla, V., Lo Re, V., Dorey-Stein, Z., Kidwai, F., Scotch, M., Womack, J., Justice, A., and Brandt, C. The Yale cTAKES extensions for document classification: architecture and application. *Journal of the American Medical Informatics Association : JAMIA 18* (2011), 614–620.

[17] Garla, V. N., and Brandt, C. Ontology-guided feature engineering for clinical text classification. *Journal of Biomedical Informatics 45*, 5 (2012), 992–998.

[18] Han, B., Obradovic, Z., Hu, Z. Z., Wu, C. H., and Vucetic, S. Substring selection for biomedical document classification. *Bioinformatics 22*, 17 (2006), 2136–2142.

[19] Hersh, W., Buckley, C., Leone, T., and Hickarn, D. OHSUMED : An Interactive Retrieval Evaluation New Large Test Collection for Research. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval* (1994), pp. 192–201.

[20] Khan, A., Baharudin, B., Lee, L. H., and Khan, K. A Review of Machine Learning Algorithms for Text-Documents Classification. *Journal of Advances in Information Technology 1*, 1 (2010), 4–20.

[21] Korde, V., and Mahender, C. N. Text Classification and Classifiers: A Survey. *International Journal of Artificial Intelligence & Applications 3*, 2 (2012), 85–99.

[22] Lewis, D. D. Reuters-21578, distribution 1.0. Available from http://www.daviddlewis.com/resources/testcollections/reuters21578/.

[23] Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research 5* (2004), 361–397.

[24] Li, Y., and Jain, A. Classification of text documents. *The Computer Journal 41*, 8 (1998), 537–546.

[25] Manevitz, L. M., and Yousef, M. One-Class SVMs for Document Classification. *Journal of Machine Learning Research 2* (2001), 139–154.

[26] Papadatos, G., van Westen, G., Croset, S., Santos, R., Trubian, S., and Overington, J. P. A document classifier for medicinal chemistry publications trained on the ChEMBL corpus. *Journal of Cheminformatics 6*, 1 (2014), 40.

[27] Polettini, N. The Vector Space Model in Information Retrieval - Term Weighting Problem Local Term-Weighting, 2004.

[28] Saeys, Y., Inza, I. n., and Larrañaga, P. A review of feature selection techniques in bioinformatics. *Bioinformatics 23*, 19 (2007), 2507–2517.

[29] Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. A Bayesian approach to filtering junk e-mail. *Learning for Text Categorization: Papers from the AAAI Workshop* (1998).

[30] Schneider, G., Clematide, S., and Rinaldi, F. Detection of interaction articles and experimental methods in biomedical literature. *BMC Bioinformatics 12*, Suppl 8 (2011), S13.

[31] Sebastiani, F. Machine Learning in Automated Text Categorization. *ACM Computing Surveys 34*, 1 (2001), 1–47.

[32] Seymour, E., Damle, R., Sette, A., and Peters, B. Cost sensitive hierarchical document classification to triage PubMed abstracts for manual curation. *BMC Bioinformatics 12* (2011), 482.

[33] Shatkay, H., Brady, S., and Wong, A. Text as data: Using text-based features for proteins representation and for computational prediction of their characteristics. *Methods 74* (2015), 54–64.

[34] Shatkay, H., Chen, N., and Blostein, D. Integrating image data into biomedical text categorization. *Bioinformatics 22*, 14 (2006), e446–e453.

[35] Trieschnigg, D., Pezik, P., Lee, V., de Jong, F., Kraaij, W., and Rebholz-Schuhmann, D. MeSH Up: effective MeSH text classification for improved document retrieval. *Bioinformatics 25*, 11 (2009), 1412–1418.

[36] Tsai, R. T.-H., Hung, H.-C., Dai, H.-J., Lin, Y.-W., and Hsu, W.-L. Exploiting likely-positive and unlabeled data to improve the identification of protein-protein interaction articles. *BMC bioinformatics 9*, Suppl 1 (2008), S3.

[37] Wang, H., Huang, M., Ding, S., and Zhu, X. Exploiting and integrating rich features for biological literature classification. *BMC bioinformatics 9*, Suppl 3 (2008), S4.

[38] Wang, P., Morgan, A. a., Zhang, Q., Sette, A., and Peters, B. Automating document classification for the Immune Epitope Database. *BMC bioinformatics 8* (2007), 269.

[39] Wang, X., Rak, R., Restificar, A., Nobata, C., Rupp, C., Batista-Navarro, R. T. B., Nawaz, R., and Ananiadou, S. Detecting experimental techniques and selecting relevant documents for protein-protein interactions from biomedical literature. *BMC Bioinformatics 12*, Suppl 8 (2011), S11.

[40] Yang, H., Nenadic, G., and Keane, J. a. Identification of transcription factor contexts in literature using machine learning approaches. *BMC bioinformatics 9*, Suppl 3 (2008), S11.

[41] Yang, Y., and Chute, C. G. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems 12*, 3 (1994), 252–277.

[42] Yeniterzi, S., and Sezerman, U. EnzyMiner: automatic identification of protein level mutations and their impact on target enzymes from PubMed abstracts. *BMC bioinformatics 10*, Suppl 8 (2009), S2.

[43] Yin, L., Xu, G., Torii, M., Niu, Z., Maisog, J. M., Wu, C., Hu, Z., and Liu, H. Document classification for mining host pathogen protein-protein interactions. *Artificial Intelligence in Medicine 49*, 3 (2010), 155–160.

[44] Yoon, Y., and Lee, G. G. Subcellular localization prediction through boosting association rules. *IEEE/ACM Transactions on Computational Biology and Bioinformatics 9*, 2 (2012), 609–618.