Cross-Domain Sentiment Classification via Polarity-Driven State Transitions in a Markov Model

Giacomo Domeniconi, Gianluca Moro, Andrea Pagliarani(✉), and Roberto Pasolini

DISI, Università Degli Studi di Bologna, Via Venezia 52, Cesena, Italy
{giacomo.domeniconi,gianluca.moro,andrea.pagliarani12, roberto.pasolini}@unibo.it

Abstract. Nowadays understanding people's opinions is the way to success, whatever the goal. Sentiment classification automates this task, assigning a positive, negative or neutral polarity to free text concerning services, products, TV programs, and so on. Learning accurate models requires a considerable effort from human experts that have to properly label text data. To reduce this burden, cross-domain approaches are advisable in real cases and transfer learning between source and target domains is usually demanded due to language heterogeneity. This paper introduces some variants of our previous work [1], where both transfer learning and sentiment classification are performed by means of a Markov model. While document splitting into sentences does not perform well on common benchmark, using polarity-bearing terms to drive the classification process shows encouraging results, given that our Markov model only considers single terms without further context information.

Keywords: Transfer learning · Sentiment classification · Markov chain · Opinion mining · Polarity-bearing terms · Sentence classification

1 Introduction

When an understanding is required of whether a plain text document has a positive, negative or neutral orientation, *sentiment classification* is involved. This supervised approach learns a model from a training set of documents, labeled with the categories (or classes) we are interested in, then applies it to the test set whose sentiment orientation has to be discovered. Sentiment classification differs from text categorization in the set of classes to be considered: whereas the former usually relies on the same labels (i.e. positive, negative and possibly neutral), in the latter topics are typically what we are interested in and they can range from music to games, to literature, to art, and so on and so forth. However, in both tasks the classification accuracy depends on how much the test documents reproduce the patterns emerged when learning the model on the training set.

Giacomo Domeniconi—This work was partially supported by the project "Toreador".

In order to achieve this goal, the most natural approach consists in considering as training set documents belonging to the same domain of those to be classified. The just outlined course of action, referred in literature as *in-domain*, is the most profitable in terms of accuracy, because documents within a unique domain are likely to be lexically and semantically similar. Nevertheless, the implicit assumption in-domain classification relies on in order to build a model, is to always have labeled documents from the same domain of the text set whose sentiment orientation is required to be predicted. This is quite difficult in practice due to some reasons: on the one hand real text sets, like for instance Facebook posts, tweets, discussions in fora, are usually unlabeled. On the other hand, labeling text sets is an onerous activity if manually performed by human experts.

Therefore *cross-domain* approaches are attractive in real cases, because if a model has been built on a certain domain, its reuse in a different domain is definitely desirable due to the aforementioned reasons. For example, let us suppose to have learned a model on a set of reviews about movies and now to be interested in understanding people's thoughts about kitchen appliances. Instead of manually labeling a part of the kitchen appliances reviews and building a different model from scratch, we would exploit the one we already have. However, the test documents might not reflect the regularity of the training set because of language heterogeneity. In fact if a movie review is set to include terms like "amusing" or "boring", a kitchen appliance review is more likely to contain "clean" or "broken". Hence, a *transfer learning* phase is typically demanded to bridge the inter-domain gap.

Many methods have been proposed in literature to transfer knowledge across domains, by either adapting source data to the target domain or representing both in a common space, using approaches such as for example feature expansion and clustering. Remarkable levels of accuracy are reported in experimental evaluations of these methods on topic [2–4] and sentiment classification [5–7]. Given their complexity, these methods often have the drawback of requiring potentially burdensome parameter tuning in order to yield good results in real use cases: at this extent, [8] presents a more straightforward method which obtains comparable experimental results with fixed parameter values.

In our previous work [1], transfer learning was performed along with sentiment classification by folding terms from both source and target domains into the same graph, characterized by a vertex for each term and an edge among terms occurring together in documents. It deserves to be noted that classes were also included into our representation. In this way, since the graph can easily be interpreted as a Markov chain, information flows step by step from source specific terms to target specific ones and reaches categories, allowing both transfer learning and sentiment classification.

In this paper we extend the aforementioned approach with two variants, namely, considering documents as aggregates of sentences rather than as a whole block and using polarity-bearing terms in order to drive state transitions in the Markov chain. The first proposal deals with the idea that not every sentence bears the same sentiment orientation of the document where it appears. Thus, classification is performed sentence by sentence and then results are combined in order to assign the final label to the document. Instead, the second variant aims to classify document sentiment through polarity-bearing terms. Whereas in the basic Markov chain method state transitions are only proportional to the semantic relationships between terms, now they also are function of the terms capability in predicting category. This means that polarity-bearing terms actually are those that contribute the most to the classification process.

Experiments have been performed to compare the underlying variants with the previous version of our Markov chain method. The same benchmark text sets have been chosen to assess performance in 2-classes (i.e. positive and negative) in-domain and cross-domain sentiment classification. On the one hand, results obtained by splitting documents into sentences are unsatisfying; on the other hand, the outcome with polarity-driven state transitions is comparable with both our basic algorithm and other works. Besides preserving the same benefits of our foregoing Markov method, the latter technique improves the way classification is carried out, that is, by taking advantage of polarity-bearing terms. Results are encouraging, especially if considering that the model currently takes only single terms into account, without relying on any kind of context information.

The rest of the paper is organized as follows. Section 2 analyzes the literature about transfer learning, sentiment classification and Markov chains. Section 3 first of all recaps our preceding Markov chain based approach, then explains the variants we introduce in this paper. Section 4 describes the experiments and compares the outcome with other works. Finally, Sect. 5 sums results up and outlines future work.

2 Related Work

Transfer learning generally entails learning knowledge from a *source* domain and using it in a *target* domain. Specifically, *cross-domain* methods are used to handle data of a target domain where labeled instances are only available in a source domain, similar but not equal to the target one. While these methods are used in image matching [9], genomic prediction [10] and many other contexts, classification of text documents by either topic or sentiment is perhaps their most common application. Two major approaches can be distinguished in cross-domain classification [11]: *instance-transfer* directly adjusts source instances to the target domain, while *feature-representation-transfer* maps features of both domains to a different common space. In text categorization by topic, transfer learning has been fulfilled in some ways, for example by clustering together documents and words [2], by extending *probabilistic latent semantic analysis* also to unlabeled instances [3], by extracting latent words and topics, both common and domain specific [4], by iteratively refining target categories representation without a burdensome parameter tuning [8,12].

Apart from the aforementioned, a number of different techniques have been developed solely for sentiment classification. For example, [13] draw on information retrieval methods for feature extraction and to build a scoring function based on words found in positive and negative reviews. In [5,6], a dictionary containing commonly used words in expressing sentiment is employed to label a portion of informative examples from a given domain, in order to reduce the labeling effort and to use the labeled documents as training set for a supervised classifier. Further, lexical information about associations between words and classes can be exploited and refined for specific domains by means of training examples to enhance accuracy [7]. Finally, term weighting could foster sentiment classification as well [14], just like it happens in other mining tasks, from the general information retrieval to specific contexts, such as prediction of gene function annotations in biology [15]. For this purpose, some researchers propose different term weighting schemes: a variant of the well-known *tf-idf* [16], a supervised scheme based on both the importance of a term in a document and the importance of a term in expressing sentiment [17], regularized entropy in combination with singular term cutting and bias term in order to reduce the over-weighting issue [18].

With reference to cross-domain setting, a bunch of methods has been attempted to address the transfer learning issue. Following works are based on some kind of supervision. In [19], some approaches are tried in order to customize a classifier to a new target domain: training on a mixture of labeled data from other domains where such data is available, possibly considering just the features observed in target domain; using multiple classifiers trained on labeled data from diverse domains; including a small amount of labeled data from target. [20] suggest the adoption of a thesaurus containing labeled data from source domain and unlabeled data from both source and target domains. [21] discover a measure of domain similarity contributing to a better domain adaptation. [22] advance a spectral feature alignment algorithm which aims to align words belonging to different domains into same clusters, by means of domain-independent terms. These clusters form a latent space which can be used to improve sentiment classification accuracy of target domain. [23] extend the *joint sentiment-topic* model by adding prior words sentiment, thanks to the modification of the topic-word Dirichlet priors. Feature and document expansion are performed through adding polarity-bearing topics to align domains.

On the other hand, document sentiment classification may be performed by using unsupervised methods as well. In this case, most features are words commonly used in expressing sentiment. For instance, an algorithm is introduced to basically evaluate mutual information between the given sentence and two words taken as reference: "excellent" and "poor" [24]. Furthermore, in another work not only a dictionary of words annotated with both their semantic polarity and their weights is built, but it also includes intensification and negation [25].

Markov chain theory, whose a brief overview can be found in Sect. 3, has been successfully applied in several text mining contexts, such as *information retrieval, sentiment analysis, text classification.*

Markov chains are particularly suitable for modeling hypertexts, which in turn can be seen as graphs, where pages or paragraphs represent states and links represent state transitions. This helps in some information retrieval tasks, because it allows discovering the possible presence of patterns when humans search for information in hypertexts [26], performing link prediction and path analysis [27] or even defining a ranking of Web pages just dealing with their hypertext structure, regardless information about page content [28].

Markov chains, in particular hidden Markov chains, have been also employed to build information retrieval systems where firstly query, document or both are expanded and secondly the most relevant documents with respect to a given query are retrieved [29,30], possibly in a *spoken document retrieval* context [31] or in the *cross-lingual area* [32]. Anyhow, to fulfil these purposes, Markov chains are exploited to model term relationships. Specifically, they are used either in a single-stage or in a multi-stage fashion, the latter just in case indirect word relationships need to be modeled as well [33].

The idea of modeling word dependencies by means of Markov chains is also pursued for *sentiment analysis*. In practice, hidden Markov models (HMMs) aim to find out opinion words (i.e. words expressing sentiment) [34], possibly trying to correlate them with particular topics [35,36]. Typically, transition probabilities and output probabilities between states are estimated by using the Baum-Welch algorithm, whereas the most likely sequence of topics and related sentiment is computed through the Viterbi algorithm. The latter algorithm also helps in *Part-of-speech (POS) tagging*, where Markov chain states not only model terms but also tags [37,38]. In fact, when a tagging for a sequence of words is demanded, the goal is to find the most likely sequence of tags for that sequence of words.

Following works are focused on *text classification*, where the most widespread approach based on Markov models consists in building a HMM for each different category. The idea is, for each given document, to evaluate the probability of being generated by each HMM, finally assigning to that document the class corresponding to the HMM maximizing this probability [39–41]. Beyond directly using HMMs to perform text categorization, they can also be exploited to model inter-cluster associations. For instance, words in documents can be clustered for dimensionality reduction purposes and each cluster can be mapped to a different Markov chain state [42]. Another interesting application is the classification of multi-page documents where, modeling each page as a different bag-of-words, a HMM can be exploited to mine correlation between documents to be classified (i.e. pages) by linking concepts in different pages [43].

3 Method Description

This Section firstly recaps the method based on the Markov chain theory we advanced in [1] to accomplish both in-domain and cross-domain sentiment classification. Then two variants of the underlying approach are proposed, which can also be combined together.

In order for non-expert readers to have a complete understanding, we would like to remind that a Markov chain is a mathematical model that is subject to transitions from one state to another in a states space $\mathcal{S}$. In particular, it is a stochastic process characterized by the so called *Markov property*, namely, future state only depends on current state, whereas it is independent of past states.

3.1 Basic Approach

Before going into details, notice that the entire algorithm can be split into three main stages, namely, the text pre-processing phase, the learning phase and the classification phase. We argue that the learning phase and the classification phase are the most innovative parts of the whole algorithm, because they accomplish both transfer learning and sentiment classification by means of only one abstraction, that is, the Markov chain.

Text Pre-processing Phase. The initial stage of the algorithm is text pre-processing. Starting from a corpus of documents written in natural language, the goal is to transform them in a more manageable, structured format.

Firstly, standard techniques are applied to the plain text, such as word tokenization, punctuation removal, number removal, case folding, stopwords removal and the Porter stemming algorithm [44]. Notice that stemming definitely helps the sentiment classification process, because words having the same morphological root are likely to be semantically similar.

The representation used for documents is the common bag-of-words, that is, a term-document matrix where each document d is seen as a multiset (i.e. bag) of words (or terms). Let $\mathcal{T} = \{t_1, t_2, \ldots, t_k\}$, where k is the cardinality of $\mathcal{T}$, be the dictionary of terms to be considered, which is typically composed of every term appearing in any document in the corpus to be analyzed. In each document d, each word t is associated to a weight w_t^d, usually independent of its position inside d. More precisely, w_t^d only depends on *term frequency* $f(t, d)$, that is, the number of occurrences of t in document d, and in particular, represents *relative frequency* $rf(t, d)$, computed as follows:

$$w_t^d = rf(t, d) = \frac{f(t, d)}{\sum\limits_{\tau \in \mathcal{T}} f(\tau, d)} \tag{1}$$

After having built the bags of words, a feature selection process is performed to fulfil a twofold goal: on the one hand, feature selection allows selecting only the most profitable terms for the classification process. On the other hand, being k higher the more the dataset to be analyzed is large, selecting only a small subset of the whole terms cuts down the computational burden required to perform both the learning phase and the classification phase.

Among the feature selection methods analyzed in [1], the one that performs better was *chi-square* χ^2, defined as in [45], which is a supervised scoring function able to find the most relevant features with respect to its ability to characterize a certain category. The ranking obtained as output is used on the one hand to select the best n features and on the other hand to change term weighting inside documents. In fact, this score $s(t)$ is a global value, stating the relevance of a certain word, whereas relative frequency, introduced by Eq. 1, is a local value only measuring the relevance of a word inside a particular document. Therefore, these values can be combined into a different term weighting to be used for the bag-of-words representation, so that the weight w_t^d comes to be

$$w_t^d = rf(t,d) \cdot s(t) \tag{2}$$

Thus, according to the Eq. 2, both factors (i.e. the global relevance and the local relevance) may be taken into account.

Learning Phase. The learning phase is the second stage of our algorithm. As in any categorization problem, the primary goal is to learn a model from a training set, so that a test set can be accordingly classified. Though, the mechanism should also allow transfer learning in cross-domain setting.

The basic idea consists in modeling term co-occurrences: the more words co-occur in documents the more their connection should be stronger. We could represent this scenario as a graph whose nodes represent words and whose edges represent the strength of the connections between them. Considering a document corpus $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ and a dictionary $\mathcal{T} = \{t_1, t_2, \ldots, t_k\}$, $A = \{a_{ij}\}$ is the set of connection weights between the term t_i and the term t_j and each a_{ij} can be computed as follows:

$$a_{ij} = a_{ji} = \sum_{d=1}^{N} w_{t_i}^d \cdot w_{t_j}^d \tag{3}$$

The same strategy could be followed to find the polarity of a certain word, unless having an external knowledge base which states that a word is intrinsically positive, negative or neutral. Co-occurrences between words and classes are modeled for each document whose polarity is given. Again, a graph whose nodes are either terms or classes and whose edges represent the strength of the connections between them is suitable to represent this relationship. In particular, given that $\mathcal{C} = \{c_1, c_2, \ldots, c_M\}$ is the set of categories and $B = \{b_{ij}\}$ is the set of edges between a term t_i and a class c_j, the strength of the relationship between a term t_i and a class c_j is augmented if t_i occurs in documents belonging to the set $D^j = \{d \in \mathcal{D} : c_d = c_j\}$.

$$b_{ij} = \sum_{d \in D^j} w_{t_i}^d \tag{4}$$

Careful readers may have noticed that the graph representing both term co-occurrences and term-class co-occurrences can be easily interpreted as a Markov chain. In fact, graph vertices are simply mapped to Markov chain nodes and graph edges are split into two directed edges (i.e. the edge linking states t_i and t_j is split into one directed edge from t_i to t_j and another directed edge from t_j to t_i). Moreover, for each state a normalization step of all outgoing arcs is enough to satisfy the probability unitarity property. Finally, the Markov property surely holds because each state only depends on directly linked states, since we evaluate co-occurrences considering just two terms (or a term and a class) at a time.

After having explained again the basic idea behind our method, we recap how the learning phase was performed in [1]. Basically, we relied on the assumption that there exist a subset of common terms between source and target domains that can act as a bridge between domain specific terms, allowing and supporting transfer learning. So, these common terms are the key to let information about classes flow from source specific terms to target specific terms, exploiting term co-occurrences, as shown in Fig. 1.

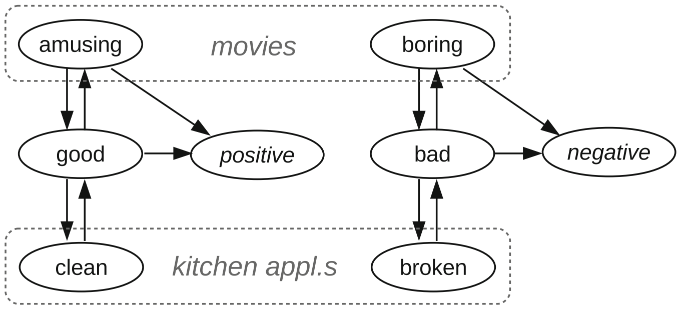

Fig. 1. Transfer learning from book-specific terms to kitchen appliances-specific terms through common terms.

We would like to point out that the just described transfer learning process is not an additional step to be added in cross-domain problems; on the contrary, it is implicit in the Markov chain mechanism and, as such, it is performed in in-domain problems as well. Obviously, if both training set and test set are extracted from the same domain, it is likely that most of the terms in test set documents already have a polarity.

Apart from transfer learning, the Markov chain we propose also fulfils the primary goal of the learning phase, that is, to build a model that can be subsequently used in the classification phase. Markov chain can be represented as a transition matrix (MCTM), composed of four logically distinct submatrices, as shown in Table 1. It is a $(k + M) \times (k + M)$ matrix, having current states as rows and future states as columns. Each entry represents a transition probability, which is computed differently depending on the type of current and future states (term or class), as described below.

Table 1. This table shows the structure of MCTM. It is composed of four submatrices, representing the transition probability that, starting from a current state (i.e. row), a future state (i.e. column) is reached. Both current states and future states can be either terms or classes.

	$t_1, \ldots, t_k$	$c_1, \ldots, c_M$
$t_1, \ldots, t_k$	$A^{'}$	$B^{'}$
$c_1, \ldots, c_M$	E	F

Let $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$ be the subsets of document corpus $\mathcal{D}$ chosen as training set and test set respectively. The set A, whose each entry is defined by Eq. 3, is rewritten as

$$a_{ij} = a_{ji} = \begin{cases} 0, & i = j \\ \sum\limits_{d \in \mathcal{D}_{train} \cup \mathcal{D}_{test}} w^d_{t_i} \cdot w^d_{t_j}, & i \neq j \end{cases} \tag{5}$$

and the set B, whose each entry is defined by Eq. 4, is rewritten as

$$b_{ij} = \sum_{d \in D^j_{train}} w^d_{t_i} \tag{6}$$

where $D^j_{train} = \{d \in \mathcal{D}_{train} : c_d = c_j\}$. The submatrices $A^{'}$ and $B^{'}$ are the normalized forms of Eqs. 5 and 6, computed so that each row of the Markov chain satisfies the probability unitarity property. Instead, each entry of the submatrices E and F looks like as follows:

$$e_{ij} = 0 \tag{7}$$

$$f_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{8}$$

Notice that E and F deal with the assumption that classes are absorbing states, which can never be left once reached.

Classification Phase. The last step of the algorithm is the classification phase. The aim is classifying test set documents by using the model learned in the previous step. According to the bag-of-words representation, a document $d_t \in \mathcal{D}_{test}$ to be classified can be expressed as follows:

$$d_t = (w^{d_t}_{t_1}, \ldots, w^{d_t}_{t_k}, c_1, \ldots, c_M) \tag{9}$$

$w^{d_t}_{t_1}, \ldots, w^{d_t}_{t_k}$ is the probability distribution representing the initial state of the Markov chain transition matrix, whereas $c_1, \ldots, c_M$ are trivially set to 0. We initially hypothesize to be in many different states (i.e. every state t_i so that $w^{d_t}_{t_i} > 0$) at the same time. Then, simulating a single step inside the Markov chain transition matrix, we obtain a posterior probability distribution not only over terms, but also over classes. In such a way, estimating the posterior probability that d_t belongs to a certain class c_i, we could assign the most likely label $c_i \in \mathcal{C}$ to d_t. The posterior probability distribution after one step in the transition matrix, starting from document d_t, is:

$$d^*_t = (w^{d^*_t}_{t_1}, \ldots, w^{d^*_t}_{t_k}, c^*_1, \ldots, c^*_M) = d_t \times MCTM \tag{10}$$

where d_t is a column vector having size $(k + M)$ and $MCTM$ is the Markov chain transition matrix, whose size is $(k + M) \times (k + M)$. At this point, the category that will be assigned to d_t is computed as follows:

$$c_{d_t} = \arg\max_{i \in C^*} c_i^* \tag{11}$$

where $C^* = \{c_1^*, \ldots, c_M^*\}$ is the posterior probability distribution over classes.

Computational Complexity. The computational complexity of our method is the time required to perform both the learning phase and the classification phase. Regarding the learning phase, the computational complexity overlaps the time needed to build the Markov chain transition matrix, say $time(MCTM)$, which is

$$\begin{aligned} time(MCTM) &= time(A) + time(B) \\ &+ time(A^{'} + B^{'}) + time(E) + time(F) \end{aligned} \tag{12}$$

Remember that A and B are the submatrices representing the state transitions having a term as current state. Similarly, E and F are the submatrices representing the state transitions having a class as current state. $time(A^{'} + B^{'})$ is the temporal length of the normalization step, mandatory in order to observe the probability unitarity property. On the other hand, E and F are simply a null and an identity matrix, requiring no computation. Thus, since time complexity depends on these factors, all should be estimated.

The only assumption we can make is that in general $|\mathcal{T}| >> |\mathcal{C}|$. The time needed to compute A is $O(\frac{|\mathcal{T}|^2}{2} \cdot (|\mathcal{D}_{train}| + |\mathcal{D}_{test}|))$, which in turn is equal to $O(|\mathcal{T}|^2 \cdot (|\mathcal{D}_{train}| + |\mathcal{D}_{test}|))$. Regarding transitions from terms to classes, building the submatrix B requires $O(|\mathcal{T}| \cdot |\mathcal{C}| \cdot |\mathcal{D}_{train}|)$ time. In sentiment classification problems we could also assume that $|\mathcal{D}| >> |\mathcal{C}|$ and, as a consequence, the previous time becomes $O(|\mathcal{T}| \cdot |\mathcal{D}_{train}|)$. The normalization step, which has to be computed one time only for both A and B, is $O(|\mathcal{T}| \cdot (|\mathcal{T}| + |\mathcal{C}|) + |\mathcal{T}| + |\mathcal{C}|) = O((|\mathcal{T}| + 1) \cdot (|\mathcal{T}| + |\mathcal{C}|))$, which can be written as $O(|\mathcal{T}|^2)$ given that $|\mathcal{T}| >> |\mathcal{C}|$. Further, building the submatrix E requires $O(|\mathcal{T}|^2)$ time, whereas for submatrix F $O(|\mathcal{T}| \cdot |\mathcal{C}|)$ time is needed, which again can be written as $O(|\mathcal{T}|)$ given that $|\mathcal{T}| >> |\mathcal{C}|$. Therefore, the overall complexity of the learning phase is

$$\begin{aligned} time(MCTM) &\simeq time(A) \\ &= O(|\mathcal{T}|^2 \cdot (|\mathcal{D}_{train}| + |\mathcal{D}_{test}|)) \end{aligned} \tag{13}$$

In the classification phase, two operations are performed for each document to be categorized: the matrix product in Eq. 10, which requires $time(MatProd)$, and the maximum computation in Eq. 11, which requires $time(Max)$. Hence, as we can see below

$$time(CLASS) = time(MatProd) + time(Max) \tag{14}$$

the classification phase requires a time that depends on the previous mentioned factors. The matrix product can be computed in $O((|\mathcal{T}| + |\mathcal{C}|)^2 \cdot |\mathcal{D}_{test}|)$ time, which can be written as $O(|\mathcal{T}|^2 \cdot |\mathcal{D}_{test}|)$ given that $|\mathcal{T}| >> |\mathcal{C}|$. On the other hand, the maximum requires $O(|\mathcal{C}| \cdot |\mathcal{D}_{test}|)$ time. Since the assumption that $|\mathcal{T}| >> |\mathcal{C}|$ still holds, the complexity of the classification phase can be approximated by the calculus of the matrix product.

Lastly, the overall complexity of our algorithm, say $time(Algorithm)$, is as follows:

$$\begin{aligned} time(Algorithm) &= time(MCTM) + time(CLASS) \\ &\simeq time(MCTM) = O(|\mathcal{T}|^2 \cdot (|\mathcal{D}_{train}| + |\mathcal{D}_{test}|)) \end{aligned} \tag{15}$$

This complexity is comparable with the best performing state of the art methods.

3.2 Document Splitting into Sentences

The first variant we advance in this work consists in considering the sentence granularity rather than the document granularity. In fact documents can be composed of many sentences, possibly characterized by a different sentiment orientation. It is not seldom to encounter documents expressing positive (resp. negative) opinions about a list of aspects and ending with an overall opposite sentiment summarized in few words. Examples of the underlying behavior are *"My car's steering wheel always vibrates because of [...] The seat is not so comfortable [...] But in general I like my car."* or even *"I read that book. Characters are well described, their psychological profile is meticulously portrayed. However, the plot is definitely boring and I always fall asleep while reading!"*

During the text pre-processing phase, documents are split into sentences using the set of characters $Tok = \{., ;, !, ?\}$ as tokenizers. In this process, we should be careful of some exceptions that can occur and invalidate the splitting. We only handle the most straightforward cases, like for example *Ph.D.*, *Dr.*, websites and emails. Anyway we are aware that, depending on the training set, many nontrivial cases could negatively affect the splitting.

In this context, co-occurrences between terms are no longer to be evaluated inside the same document, but within the same sentence. More formally, considering each document d_z as a set of sentences $d_z = \{p_1^z, p_2^z, \ldots, p_h^z\}$, Eq. 3 can be rewritten as follows:

$$a_{ij} = a_{ji} = \sum_{d \in \mathcal{D}} \sum_{p \in d} w_{t_i}^p \cdot w_{t_j}^p \tag{16}$$

Consequently, Eq. 5 becomes

$$a_{ij} = a_{ji} = \begin{cases} 0, & i = j \\ \sum\limits_{d \in \mathcal{D}_{train} \cup \mathcal{D}_{test}} \sum\limits_{p \in d} w_{t_i}^p \cdot w_{t_j}^p, & i \neq j \end{cases} \tag{17}$$

Similarly, the transition from a term to a class in Eq. 4 can be rewritten as

$$b_{ij} = \sum_{d \in \mathcal{D}} \sum_{p^j \in d} w_{t_i}^{p^j} \tag{18}$$

where $p^j = \{p_t \in d : c_{p_t} = c_j\}$. Likewise, Eq. 6 comes to be

$$b_{ij} = \sum_{d \in \mathcal{D}_{train}} \sum_{p^j \in d} w_{t_i}^{p^j} \tag{19}$$

In the classification phase, we modify Eqs. 10 and 11 to label the single sentences inside a document rather than the document itself:

$$p_t^* = (w_{t_1}^{p_t^*}, \ldots, w_{t_k}^{p_t^*}, c_1^*, \ldots, c_M^*) = p_t \times MCTM \tag{20}$$

$$c_{p_t} = \arg\max_{i \in C^*} c_i^* \tag{21}$$

where p_t is a column vector having size $(k + M)$.

The output labels for each sentence are finally combined by voting in order to obtain the final category for the document to be classified:

$$c_d = \arg\max_{i \in \mathcal{C}} |c_i^d| \tag{22}$$

where $|c_i^d| = |\{p_t \in d : c_{p_t} = c_i\}|$. The computational complexity is now function of the number of sentences in the corpus, rather than of the number of documents as in the basic version.

3.3 Polarity-Driven State Transitions

A second alternative, orthogonal to the previous one, has been developed to establish how much a term should be linked with the others and with classes. To this purpose we have to take into account that the probability that the current state has at time t will be redistributed to the other states at time $t + 1$. So far this takes place based on co-occurrences, as stated by Eqs. 5 and 6. Although it might seem reasonable, it is not enough because the capability of both the current state and the future state in discriminating among categories is completely overlooked during state transitions. Some issues related to this behavior could occur. For example, if the current state is not well polarized, not only it will not be able to distinguish among classes, but it will likely be connected to terms having conflicting sentiment. On the other hand, if a term semantically related to the current state is not well polarized, it should not be selected as future state because it will not be useful for classification.

In order to solve the just explained issues, we focus on what it should take place during state transitions. The intuition is that the more the current state is capable of discriminating among categories (i.e. it is polarity-bearing) the more its probability will be given to classes. The remaining part will be distributed to the other terms in a proportional way not only to the semantic relationship between the current state and the future state, but also to the capability of the latter in distinguishing among classes. Everything we need to fulfil this twofold goal already is in our basic version of the Markov chain. In fact, the capability f_i of a term t_i in discriminating among categories can be defined as:

$$f_i = \frac{|b_{i+} - b_{i-}|}{b_{i+} + b_{i-}} \tag{23}$$

where b_{ij} is what has been defined by Eq. 6 and we can notice that $0 \leq f_i \leq 1$. In other words, f_i is the portion of probability that t_i will redistribute to classes in a proportional way to the values computed by Eq. 6. This means that polarity-bearing terms are those that contribute the most to the classification process. The remaining $(1 - f_i)$ will be split among terms according to the following relation:

$$a_{ij} = \begin{cases} 0, & i = j \\ f_j \cdot \sum_{d \in \mathcal{D}_{train} \cup \mathcal{D}_{test}} w_{t_i}^d \cdot w_{t_j}^d, & i \neq j \end{cases} \tag{24}$$

Notice that the transition probability depends not only on the semantic relationship among terms, but also on the capability of the destination term in detecting categories.

We would like to remind that, according to Eq. 9, when classifying a document d_t the initial state of the Markov chain was represented by $w_{t_1}^{d_t}, \ldots, w_{t_k}^{d_t}$. Each weight w_{t_i} has to be multiplied by f_i as well, since only polarity-bearing terms should drive sentiment classification, spreading their probability when performing a step in the Markov chain. The overall computational complexity is aligned with that of the aforementioned basic approach.

4 Experiments

The Markov chain based methods have been implemented in a framework entirely written in Java. Algorithms performance has been evaluated through the comparison with *Spectral feature alignment* (*SFA*) by Pan et al. [22] and *Joint sentiment-topic model with polarity-bearing topics* (*PBT*) by He et al. [23], which, to the best of our knowledge, currently are the two best performing approaches in cross-domain sentiment classification.

We used common benchmark datasets to be able to compare results, namely, a collection of Amazon[1] reviews about four domains: Book (B), DVD (D), Electronics (E) and Kitchen appliances (K). Each domain contains 1000 positive and 1000 negative reviews written in English. The text pre-processing phase described in Sect. 3.1 is applied to convert plain text into the bag-of-words representation, possibly splitting documents into sentences when dealing with the variant introduced in Sect. 3.2. Before the learning phase and the classification phase, we perform feature selection by means of χ^2, which turned out to be the best performing technique for this purpose in [1].

Performance of every presented variant is shown below and compared with the state of the art. Differently, the Kitchen domain is ruled out from the analysis, in order for the results to be comparable with those reported in our previous work.

4.1 Setup and Results

From now on we will use MC_S when referring to the Markov chain variant characterized by splitting documents into sentences, MC_P for the polarity-driven transitions one, MC_{SP} for the combination between them, whereas MC_B indicates the basic approach.

In order to compare performance with MC_B, we replicate the same experiment that gave the best outcome in our previous work. Therefore, the training set is composed of 1600 documents and the test set of 400 documents. The best 250 features are selected in the text pre-processing phase by means of χ^2 scoring function. The goodness of results is measured by accuracy, averaged over 10 random source-target splits and evaluated for each particular source-target combination, namely $B \to D$, $D \to B$, $B \to E$, $E \to B$, $D \to E$, $E \to D$, even including in-domain configurations, such as $B \to B$, $D \to D$, $E \to E$.

As we can notice in Fig. 2, both the variants relying on sentence splitting (i.e. MC_S and MC_{SP}) do not perform well. The reason for this outcome is to be found in the learning phase, where the polarity of each sentence should be taken into account, as stated by Eq. 18. However, in the Amazon corpus we are only aware of the whole document polarity and not of the sentiment at sentence level. Consequently we had to make the strong assumption that each sentence

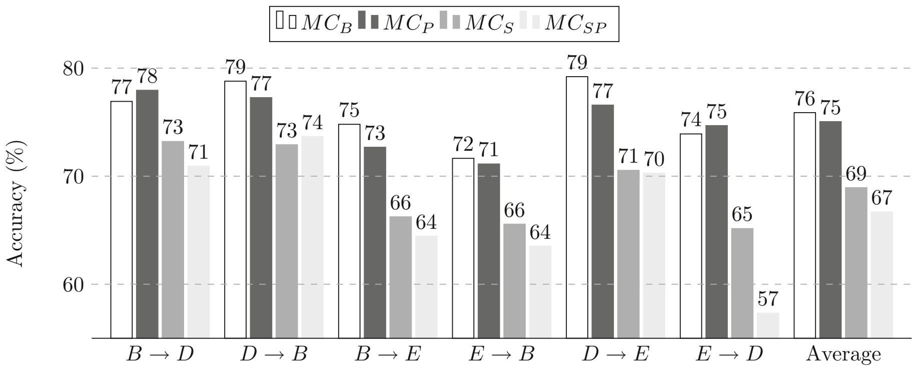

Fig. 2. Cross-domain classification by comparing all the proposed Markov chain based variants. The best 250 features are selected in accordance with the score output by χ^2.

[1] www.amazon.com.

in a training set document has the same sentiment of the document itself, even if this is definitely false in general. This could trivially bring to erroneously consider sentiment at term level and, as a consequence, to bad performance.

On the other hand, looking at Fig. 3 we can see a qualitative comparison between MC_P and MC_B. The reported examples show that polarity-driven state transitions could be helpful for the classification process when there are some polarity-bearing terms within the document to be classified. In fact, in such cases MC_P classifier is much more confident with its prediction than MC_B. This could also bring (as in example B) to correctly predict test instances failed by the basic classifier.

Anyway even if MC_P is able to take advantage of polarity-bearing terms, its accuracy does not outperform that of MC_B. This outcome could be explained considering that there is no constraint that forces terms to redistribute their probability to others having the same sentiment. Moreover, a document not con-

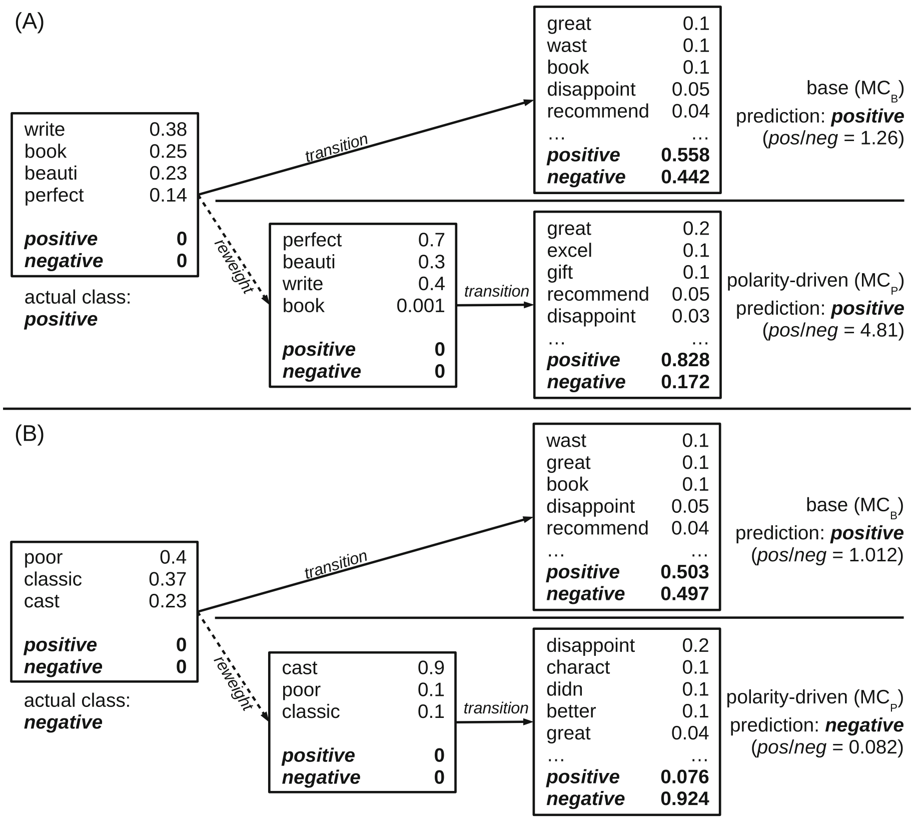

Fig. 3. Two examples of documents (A and B), represented as selected features with associated weights, classified by using either MC_B or MC_P. Each of the rightmost boxes only shows the 5 terms with the highest weights after the transition.

taining most terms bearing its own polarity is likely to be misclassified because, although terms are semantically related, they separately contribute to classification. This happens for example when a positive (resp. negative) document expresses opposite opinions about some aspects before summarizing in few words its overall polarity. Actually the context inside the document to be classified is neglected, because the classification starts assuming to be in many different states at the same time corresponding to the terms occurring in the document itself. Then the step by step evolution of each state is independent of the others; it is only determined by the semantic relationships between each term and the others, learned when training the model.

Table 2 shows a comparison between our Markov chain based methods and other works, namely SFA and PBT. Whereas MC_S and MC_{SP} are far away from the state of the art, MC_B and MC_P achieve comparable results, despite both SFA and PBT perform better on average. On the other side, we would like to emphasize that our algorithms require much fewer features than the others, i.e. 250 with respect to the 2000 needed by PBT and the more than 470000 needed by SFA. Therefore, since the computational complexity quadratically grows with the number of features in all methods, the convergence of both MC_B and MC_P is supposed to be dramatically faster than that of SFA and PBT.

Lastly, we can see that similar considerations can be done in an in-domain setting. Nothing needs to be changed in our methods to perform in-domain sentiment classification, whereas other works use standard classifiers completely bypassing the transfer learning phase.

Table 2. Performance of all the Markov chain based methods in both in-domain and cross-domain sentiment classification, compared with other works. For each dataset, the best accuracy is in bold.

Domain(s)	Other methods		Markov chain method variants			
	SFA	PBT	MC_S	MC_{SP}	MC_B	MC_P
Cross-domain experiments ($source \rightarrow target$)						
$B \rightarrow D$	**81.50** %	81.00 %	73.21 %	70.92 %	76.92 %	77.95 %
$D \rightarrow B$	78.00 %	**79.00** %	72.91 %	73.67 %	78.79 %	77.27 %
$B \rightarrow E$	72.50 %	**78.00** %	66.24 %	64.45 %	74.80 %	72.68 %
$E \rightarrow B$	**75.00** %	73.50 %	65.56 %	63.52 %	71.65 %	71.13 %
$D \rightarrow E$	77.00 %	79.00 %	70.54 %	70.28 %	**79.21** %	76.58 %
$E \rightarrow D$	**77.50** %	76.00 %	65.15 %	57.32 %	73.91 %	74.68 %
Average	76.92 %	77.75 %	68.94 %	66.69 %	75.88 %	75.05 %
In-domain experiments						
$B \rightarrow B$	**81.40** %	79.96 %	73.72 %	72.45 %	76.77 %	77.78 %
$D \rightarrow D$	82.55 %	81.32 %	78.34 %	79.60 %	**83.50** %	82.49 %
$E \rightarrow E$	**84.60** %	83.61 %	77.78 %	78.54 %	80.90 %	79.55 %
Average	82.85 %	81.63 %	76.61 %	76.86 %	80.39 %	79.94 %

5 Conclusions

In this work we presented some variants of the Markov chain based method already advanced in [1] to accomplish sentiment classification in both in-domain and cross-domain settings.

The first one consists in considering documents at the sentence granularity instead of perceiving them as a whole. Results dealing with this expedient are not good, probably because we made the strong assumption that, when learning the model, sentences have the same polarity of the document including them. A possible walk around is introducing a threshold parameter that establishes the probability that a sentence has the same polarity of the document where it appears. Another viable improvement consists in changing the way sentence labels are folded together to output the final category for the test document. Moreover, when a review is long, the final part usually bears the same sentiment of the entire text, because it contains a summary of the author's thought. On the contrary when it is short, it is likely that the author immediately summarizes his opinion without using terms bearing conflicting sentiment. In both cases, taking only the last few sentences into account could be profitable. A further alternative is using document splitting into sentences just to limit co-occurrences among terms rather than to change the connection between terms and classes.

The second illustrated approach addresses the problem of steering the probability each state has at time t to other states at time $t + 1$ that are capable of discriminating among categories. Although being comparable, this variant does not outperform our basic Markov chain approach. The outcome is somewhat surprising if we think that the classification is driven by polarity-bearing terms. The reason is probably to be found in the fact that there is no constraint that forces terms to redistribute their probability to others having the same sentiment. To overcome this problem we might limit terms spreading their probability to others in a way such that the more the current term is able in discriminating among categories the more it should give its probability to other terms having the same sentiment orientation.

Apart from the mentioned flaws, both the presented variants preserve all the advantages of our basic Markov chain based method. Indeed, they not only act as classifiers, but also allow transfer learning from source domain to target domain in cross-domain problems. The polarity-driven state transitions variant achieves comparable performance in terms of accuracy with the state of the art, whereas the document splitting based ones are outperformed, perhaps due to the strong assumption made. On the other side, all the introduced techniques require lower parameter tuning than previous works. Furthermore, in spite of having a comparable computational complexity, growing quadratically with the number of features, much fewer terms are demanded to obtain good accuracy.

Future work should aim to improve the algorithm effectiveness. In addition to the specific aspects proposed to enhance the particular variants, we believe that the hypotheses we rely on should be better analyzed. For example, the algorithm could suffer from the assumption to be in many different states at the same time, made when a test document is required to be classified. In fact the step by step evolution of each state is independent of the others and consequently context information ends up being overlooked. A possible way to walk could take into account ngram features rather than just unigrams. Another option in order to introduce context information is to consider grammatical relations among terms for the sake of detecting patterns.

After having enhanced the algorithms accuracy, performance in a 3-classes setting (i.e. adding the neutral category) could also be tested. On the other hand, due to their generality, our methods might be applied as is in text categorization problems. Finally, their applicability could be easily extended to other languages, because they only depend on co-occurrences among terms.

References

1. Domeniconi, G., Moro, G., Pagliarani, A., Pasolini, R.: Markov chain based method for in-domain and cross-domain sentiment classification. In: Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, pp. 127–137 (2015)
2. Dai, W., Xue, G.-R., Yang, Q., Yu, Y.:Co-clustering based classification for out-of-domain documents. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 210–219. ACM (2007)
3. Xue, G.-R., Dai, W., Yang, Q., Yu, Y.: Topic-bridged PLSA for cross-domain text classification. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and development in Information Retrieval, pp. 627–634. ACM (2008)
4. Li, L., Jin, X., Long, M.: Topic correlation analysis for cross-domain text classification. In: AAAI (2012)
5. Tan, S., Wang, Y., Cheng, X.: Combining learn-based and lexicon-based techniques for sentiment detection without using labeled examples. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 743–744. ACM (2008)
6. Qiu, L., Zhang, W., Hu, C., Zhao, K.: Selc: a self-supervised model for sentiment classification. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 929–936. ACM (2009)
7. Melville, P., Gryc, W., Lawrence, R.D.: Sentiment analysis of blogs by combining lexical knowledge with text classification. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1275–1284. ACM (2009)
8. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: Cross-domain text classification through iterative refining of target categories representations. In: Proceedings of the 6th International Conference on Knowledge Discovery and Information Retrieval (2014)
9. Shrivastava, A., Malisiewicz, T., Gupta, A., Efros, A.A.: Data-driven visual similarity for cross-domain image matching. ACM Trans. Graph. (TOG) **30**, 154 (2011)
10. Domeniconi, G., Masseroli, M., Moro, G., Pinoli, P.: Cross-organism learning method to discover new gene functionalities. Comput. Methods Programs Biomed. **126**, 20–34 (2016)
11. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010)

12. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: Iterative refining of category profiles for nearest centroid cross-domain text classification. In: Fred, A., Dietz, J.L.G., Aveiro, D., Liu, K., Filipe, J. (eds.) IC3K 2014. CCIS, vol. 553, pp. 50–67. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25840-9_4
13. Dave, K., Lawrence, S., Pennock, D.M.: Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: Proceedings of the 12th International Conference on World Wide Web, pp. 519–528. ACM (2003)
14. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: A comparison of term weighting schemes for text classification and sentiment analysis with a supervised variant of tf.idf. In: Helfert, M., Holzinger, A., Belo, O., Francalanci, C. (eds.) DATA 2015. CCIS, vol. 584, pp. 39–58. Springer, Heidelberg (2016). doi:10.1007/978-3-319-30162-4_4
15. Domeniconi, G., Masseroli, M., Moro, G., Pinoli, P.: Random perturbations of term weighted gene ontology annotations for discovering gene unknown functionalities. In: Fred, A., Dietz, J.L.G., Aveiro, D., Liu, K., Filipe, J. (eds.) IC3K 2014. CCIS, vol. 553, pp. 181–197. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25840-9_12
16. Paltoglou, G., Thelwall, M.: A study of information retrieval weighting schemes for sentiment analysis. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 1386–1395. Association for Computational Linguistics (2010)
17. Deng, Z.-H., Luo, K.-H., Yu, H.-L.: A study of supervised term weighting scheme for sentiment analysis. Expert Syst. Appl. **41**(7), 3506–3513 (2014)
18. Wu, H., Gu, X.: Reducing over-weighting in supervised term weighting for sentiment analysis. In: COLING, pp. 1322–1330 (2014)
19. Aue, A., Gamon, M.: Customizing sentiment classifiers to new domains: A case study. In: Proceedings of Recent Advances in Natural Language Processing (RANLP), vol. 1, pp. 2–1 (2005)
20. Bollegala, D., Weir, D., Carroll, J.: Cross-domain sentiment classification using a sentiment sensitive thesaurus. IEEE Trans. Knowl. Data Eng **25**(8), 1719–1731 (2013)
21. Blitzer, J., Dredze, M., Pereira, F., et al.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: ACL, vol. 7, pp. 440–447 (2007)
22. Pan, S.J., Ni, X., Sun, J.-T., Yang, Q., Chen, Z.: Cross-domain sentiment classification via spectral feature alignment. In: Proceedings of the 19th International Conference on World wide web, pp. 751–760. ACM (2010)
23. He, Y., Lin, C., Alani, H.: Automatically extracting polarity-bearing topics for cross-domain sentiment classification. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 123–131. Association for Computational Linguistics (2011)
24. Turney, P.D.: Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 417–424. Association for Computational Linguistics (2002)
25. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Comput. linguist. **37**(2), 267–307 (2011)
26. Qiu, L.: Markov models of search state patterns in a hypertext information retrieval system. J. Am. Soc. Inf. Sci. **44**(7), 413–427 (1993)
27. Sarukkai, R.R.: Link prediction and path analysis using markov chains. Comput. Netw. **33**(1), 377–386 (2000)

28. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. Technical Report, Stanford University (1999)
29. Mittendorf, E., Schäuble, P.: Document and passage retrieval based on hidden Markov models. In: Croft, B.W., van Rijsbergen, C.J. (eds.) SIGIR 1994, pp. 318–327. Springer, Heidelberg (1994)
30. Miller, D.R., Leek, T., Schwartz, R.M.: A hidden Markov model information retrieval system. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 214–221. ACM (1999)
31. Pan, Y.-C., Lee, H.-Y., Lee, L.-S.: Interactive spoken document retrieval with suggested key terms ranked by a Markov decision process. IEEE Trans. Audio Speech Lang. Process. **20**(2), 632–645 (2012)
32. Xu, J., Weischedel, R.: Cross-lingual information retrieval using hidden Markov models. In: Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing, Very Large Corpora: held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics, vol. 13, pp. 95–103. Association for Computational Linguistics (2000)
33. Cao, G., Nie, J.-Y., Bai, J.: Using Markov chains to exploit word relationships in information retrieval. In: Large Scale Semantic Access to Content (Text, Image, Video, and Sound), pp. 388–402. Le Centre de Hautes Etudes Internationales D'Informatique Documentaire (2007)
34. Li, F., Huang, M., Zhu, X.: Sentiment analysis with global topics and local dependency. In: AAAI, vol. 10, pp. 1371–1376 (2010)
35. Mei, Q., Ling, X., Wondra, M., Su, H., Zhai, C.: Topic sentiment mixture: modeling facets and opinions in weblogs. In: Proceedings of the 16th International Conference on World Wide Web, pp. 171–180. ACM (2007)
36. Jo, Y., Oh, A.H.: Aspect and sentiment unification model for online review analysis. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, pp. 815–824. ACM (2011)
37. Jin, W., Ho, H.H., Srihari, R.K.: Opinionminer: a novel machine learning system for web opinion mining and extraction. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1195–1204. ACM (2009)
38. Nasukawa, T., Yi, J.: Sentiment analysis: capturing favorability using natural language processing. In: Proceedings of the 2nd International Conference on Knowledge Capture, pp. 70–77. ACM (2003)
39. Yi, K., Beheshti, J.: A hidden Markov model-based text classification of medical documents. J. Inf. Sci. **35**, 67–81 (2008)
40. Xu, R., Supekar, K., Huang, Y., Das, A., Garber, A.: Combining text classification and hidden Markov modeling techniques for structuring randomized clinical trial abstracts. In: AMIA Annual Symposium Proceedings 2006, p. 824. American Medical Informatics Association (2006)
41. Yi, K., Beheshti, J.: A text categorization model based on hidden Markov models. In: Proceedings of the Annual Conference of CAIS/Actes du congrès annuel de l'ACSI (2013)
42. Li, F., Dong, T.: Text categorization based on semantic cluster-hidden markov models. In: Tan, Y., Shi, Y., Mo, H. (eds.) ICSI 2013. LNCS, vol. 7929, pp. 200–207. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38715-9_24

43. Frasconi, P., Soda, G., Vullo, A.: Hidden Markov models for text categorization in multi-page documents. J. Intell. Inf. Syst. **18**(2–3), 195–217 (2002)
44. Porter, M.F.: An algorithm for suffix stripping. Program **14**(3), 130–137 (1980)
45. Domeniconi, G., Moro, G., Pasolini, R., Sartori, C.: A study on term weighting for text categorization: a novel supervised variant of tf.idf. In: Proceedings of the 4th International Conference on Data Management Technologies and Applications (2015)