Applied Intelligence (2018) 48:2967–2982
https://doi.org/10.1007/s10489-017-1109-7

Hybridizing metric learning and case-based reasoning for adaptable clickbait detection

Daniel López-Sánchez[1] · Jorge Revuelta Herrero[1] · Angélica González Arrieta[1] · Juan M. Corchado[1,2]

Published online: 29 December 2017

Abstract

The term clickbait is usually used to name web contents which are specifically designed to maximize advertisement monetization, often at the expense of quality and exactitude. The rapid proliferation of this type of content has motivated researchers to develop automatic detection methods, to effectively block clickbaits in different application domains. In this paper, we introduce a novel clickbait detection method. Our approach leverages state-of-the-art techniques from the fields of deep learning and metric learning, integrating them into the Case-Based Reasoning methodology. This provides the model with the ability to learn-over-time, adapting to different users' criteria. Our experimental results also evidence that the proposed approach outperforms previous clickbait detection methods by a large margin.

Keywords Clickbait detection · Metric learning · Case-based reasoning · Neural networks

1 Introduction

Clickbaits are defined as web contents specifically designed to maximize advertisement monetization, often at the expense of quality and exactitude. They do so by using sensationalist headlines, aiming to attract a greater portion of clicks. Language patterns typically used by clickbaits seek to exploit the *curiosity gap*, providing enough information to trigger readers' curiosity but not enough to satisfy this curiosity without clicking on the link. However, the websites pointed by clickbait links have high bounce rates. This occurs due to the viewers' high expectations when clicking on the link and their subsequent disappointment when seeing the poor quality content delivered by the web page. Some examples of clickbait headlines are: *"His First 4 Sentences Are Interesting. The 5th Blew My Mind. And Made Me A Little Sick."*, *"This Kid Just Died. What He Left Behind Is Wondtacular"* (sic) or *"11 Things Parents Should Never Apologize For"*.

The swift proliferation of this type of content forced social networks to adopt mechanisms that limit the widespread use of clickbaits. For instance, in August 2014 Facebook announced[1] that posts using clickbait would be penalized by its news-feed ranking algorithm. To this end, they would measure bounce rates and ratios of people clicking on the content compared to people discussing and sharing it. However, this clickbait detection approach requires a certain volume of users and access to information about their behavior, thus being only applicable by social networks themselves.

In the literature, several researchers have tried to automatize the task of clickbait detection by applying different techniques from the fields of natural language processing, machine learning and pattern recognition [1–4]. Nevertheless, accuracy rates still need to be improved in order for these methods to be applicable in real world scenarios. All the approaches to clickbait detection proposed in the literature use classical classification algorithms along with hand-crafted representations of headlines. In this regard, the use of state-of-the-art techniques for text processing that recently emerged in the context of the deep learning paradigm might enable a boost in the accuracy of clickbait detection methods.

One disadvantage of current methods for clickbait detection is the amount of information they require to decide whether the headline under consideration is a clickbait. Some approaches (e.g., [4]) rely on the entire content of

✉ Daniel López-Sánchez
lope@usal.es

1 BISITE Research Group, Edificio I+D+i, University of Salamanca, 37007 Salamanca, Spain

2 Osaka Institute of Technology, Osaka, Japan

[1] http://newsroom.fb.com/news/2014/08/news-feed-fyi-click-baiting/.

web pages linked by headlines to emit a prediction. In other words, these approaches require accessing the web content linked by the headline under analysis in order to determine whether it must be considered as a clickbait. This implies that a system using these methods would need to follow the links associated to every single headline in a web page to determine whether they are clickbaits or not. Of course, a system taking this approach would incur high computational costs. As a consequence, the use of such methods in real-world applications is greatly limited.

Another aspect of clickbait detection that needs to be improved is the problem of adaptability. Due to the subjective nature of clickbaits, a single headline may be perceived differently by users with different interests or criteria. For this reason, a headline might be categorized as a clickbait by one user and as a legitimate headline by others. In this regard, automatic clickbait detection models should be able to adapt themselves to the criteria of specific users, improving the quality of their predictions over-time, as the user provides additional feedback.

In this paper, we propose applying the Case-Based Reasoning (CBR) methodology to address the problem of automatic clickbait detection. The use of this methodology will allow our system to adapt itself to the specific criteria of users, modifying its original configuration to fit the subjectivity of specific users by the incorporation of new revised cases into its Case-Base. The main innovation behind the proposed CBR system is the use of state-of-the-art techniques from the field of deep learning for the generation of a suitable case-representation. In particular, our system uses a pre-trained *word2vec* model [5] combined with a 1D-Convolutional neural network [6] as a means to generate suitable case-representations. To achieve this, the neural network is trained using a metric learning algorithm designed to minimize the distance between samples form the same category in the resulting feature space, while maximizing the distances between samples from different categories [7]. As a result, the neural network can be used to generate a case-representation space where samples (i.e., headlines) are grouped together in different clusters depending on whether they are a clickbait or not. Consequently, this representation enables the CBR system to achieve high clickbait detection accuracy rates. Notably, the proposed CBR system achieves these high accuracy rates by just analyzing headlines themselves, without the need of accessing the linked content.

The experimental results presented in this paper evidence that our approach outperforms previous clickbait detection approaches on the largest publicly available clickbait detection dataset. Also, the results prove that the flexibility and adaptability granted by the CBR methodology allow our system to accommodate to the subjective criteria of users. In other words, our system learns over time and adapts itself to the specific criteria of users who, depending on their preferences, might accept some of the clickbait content.

The rest of this paper is structured as follows. Section 2 summarizes some of the most important works on clickbait detection available in the literature. Section 3 introduces the classical methods used in the context of CBR to handle textual information. The overall architecture of the proposed framework and its building blocks are described in detail in Section 4. Section 5 reports on a series of experiments concerning the accuracy of the proposed framework, as compared with classical textual CBR methods and other alternative approaches. Finally, Section 6 discusses the experimental results obtained in the previous section and outlines future research lines.

2 Related work

Throughout the years, innumerable authors have addressed the problems of spam and fake web-content detection (see, for instance [8, 9]). However, as the authors of [1] have pointed out, clickbaits are not necessarily spam or fake. Instead, they are genuine web-pages which deliver poor quality content. For this reason, several authors have tried to address the specific problem of automatic clickbait detection [1–4, 10]. Unfortunately, direct comparison of accuracies is often not possible due to the lack of a standard dataset for clickbait detection and the constraints of the different methods. For this reason, most comparisons established in this section are at a qualitative level. Nevertheless, we decided to use the biggest publicly available clickbait dataset for our experiments, with the goal of making our results as reproducible as possible. More details about the experimental setup and the results can be found in Section 5.

Chakraborty et al. [1] recently collected the biggest clickbait dataset available in the literature, with approximately 32,000 headlines. They performed a descriptive analysis comparing clickbaits and non-clickbaits in terms of sentence structures and language usage. From their analysis, they derived a set of discriminative features and evaluated different classification algorithms trained on those features. In particular, their custom features describe the structure of sentences, word patterns, use of clickbait language and n-gram features pruned using the sub-sequence property and an APRIORY-like algorithm [11]. The evaluated classifiers were the Support Vector Machine (SVM) with Radial Basis Kernel (RBF), The Random Forest algorithm [12] and a simple Decision Tree. The method which exhibited a greater accuracy was the RBF-SVM, with a 93% classification accuracy. They also evaluated a detection method based on the set of rules used by [10] as a baseline for comparison, showing that their method outperformed this naive rule-based approach by a significant margin.

In [4], Biyani et al. proposed a categorization criteria for clickbaits (e.g. exaggerations, ambiguous, teasing...). In addition, they proposed a clickbait detection method. However, their approach is based on the textual content of web pages linked by clickbaits, thus being less widely applicable (i.e. for an automatic clickbait detection system to work, it should access the content linked by every single headline in a web page). They use a set of features which include term-frequencies (unigrams and bigrams) along with several other handcrafted features, such as binary features encoding the presence of exclamation marks, numbers and superlative adverbs. Those features were used to train a Gradient Boosting Decision Tree [13] classifier, which obtained a 0.603 F-1 score in the test set collected by the authors.

Potthast et al. [3] recently presented a method for automatic clickbait detection. In contrast with other authors, they focused on clickbaits in the social network Twitter. As a consequence, some of the features they used for classification are only available in this context (e.g. Twitter user name, mentions and hashtags...). They collected a corpus with 3,000 tweets from top Twitter publishers and evaluated a series of classical classifiers over different sets of features. As in [4], the authors propose using n-gram features extracted form the textual contents of linked web-pages. The best scoring model was a Random Forest [14] classifier with a 0.76 F-1 score. One of the most significant contributions of Potthast's work is the inclusion of a modern sentiment analysis model [15] as part of the feature extraction pipeline.

Finally, Chen et al. [2] present a descriptive analysis of the problem of clickbait in the context of online journalism. In particular, they present the phenomena of clickbait as a modern form of *tabloidization*. They also performed a descriptive comparison among basic clickbait detection approaches, and analyzed possible applications of such technology in domains such as news analysis, online content management and news aggregation.

3 Classical CBR text representations

This section reviews some of the most popular document representations and similarity measures used in the literature to enable CBR systems leverage knowledge sources in textual format. We use these methods as a baseline fo r comparison with our case-representation model, which is based on a neural network.

3.1 Word count vector

Count-based vector space models are generated based on the number of occurrences of a set of words (or n-grams) in the documents. The simplest count-based vector space model generates a vector representation where each dimension corresponds to the count of occurrences of a specific word in the document [16]. As a consequence, the dimension of document vectors is determined by the size of the dictionary (i.e., the set of words considered). If n-grams are used instead of individual words, the number of possible combination scales exponentially. Fortunately, data matrices generated by this method are often sparse, thus enabling the application of efficient classification methods and implementations designed to take advantage of this sparsity in training data [17].

When using this vector space model, the most widespread document vector similarity measure is the cosine similarity. Formally, the cosine similarity between two vectors x, y is defined as the cosine of the angle between them, and is computed as follows:

$$\cos\theta = \frac{\langle x, y\rangle}{||x||_2||y||_2} \tag{1}$$

where $\langle x, y\rangle$ is the dot-product of vector documents x and y, and $||x||_2$ is the L^2 norm of x. However, in some cases the regular Euclidean distance is used as the vector similarity measure.

3.2 Term frequency - inverse document frequency

The Term Frequency - Inverse Document Frequency (TF-IDF) vector space model is a hybrid method which combines both TF and IDF definitions in order to produce a composite weight for each term in each document. Given a set D of text documents, the TF-IDF weighting scheme [18] assigns to term t a weight in document $d \in D$ given by:

$$TF-IDF(t,d) = tf(t,d)\cdot idf(t,d) \tag{2}$$

where $tf(t,d)$ is the number of occurrences of term t in document d, and $idf(t,D)$ is the inverse document frequency, defined as:

$$idf(t,D) = \log\frac{|D|}{|\{d \in D : t \in d\}| + 1} \tag{3}$$

Intuitively, $TF-IDF_{t,d}$ assigns to term t a weight in document d that is: (1) highest when t occurs many times within a small number of documents; (2) lower when the term occurs fewer times in a document, or occurs in many documents; and (3) lowest when the term occurs in nearly all documents. At this point, we represent each document as a vector with one component corresponding to each term in the dictionary, together with a weight for each component given by (2). Dictionary terms that don't occur in a document are assigned a weight value of zero. The goal of using TF-IDF instead of the raw frequencies of occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and hence are empirically less informative than features that occur in a small fraction of the training corpus.

3.3 Latent semantic indexing

Latent Semantic Indexing [19] (LSI) is a document indexing and retrieval method which works under the assumption that a document is a random choice of words that describes a latent semantic structure. It was designed to overcome the inability of simple vector space representations to deal with two common characteristics of natural languages, namely *synonymy* and *polysemy*. Formally, it performs a truncated variant of Singular Value Decomposition (SVD) on n-gram count or TF-IDF document matrices. This is achieved by computing only the k largest singular values, where k is the desired output document dimension. Formally, given a $d \times n$ term-document matrix X (with terms as rows and documents as columns), and the desired embedding dimension k, the truncated SVD of X is given by:

$$X \approx X_k = U_k \Sigma_k V_k^T \tag{4}$$

Then, a document vector x is mapped into its k-dimensional LSI representation x_k by:

$$x_k = \Sigma_k^{-1} U_k^T x \tag{5}$$

While cosine similarities are known to perform poorly on term-document matrices, LSI combats this by providing a compact representation that merges words with strong associations in training data (i.e. similar meanings). Intuitively, a document containing the word *combat* will exhibit a high cosine similarity in the LSI space with documents containing the words *battle* and *conflict*.

3.4 Random projection

Although the Random Projection [20] (RP) algorithm was not originally designed for textual document representation, its use has progressively gained popularity in this context and it is currently present in some of the most popular toolboxes for text-topic modeling [21]. In particular, it is often used to approximate TF-IDF distances between documents in a compressed representation.

As opposed to other dimensionality reduction methods, Random Projection computes the projection matrix from a random distribution. As a consequence, the projection matrix is data independent, and no training data is required by the algorithm. Formally, the $d \times k$ projection matrix R is populated according to the sparse distribution proposed in [20]. After this, a $n \times d$ document-term matrix X (with documents as rows and terms as columns) can be randomly projected to a k-dimensional Euclidean space by:

$$X' = \frac{1}{\sqrt{k}} X R \tag{6}$$

where the projection itself can be computed in terms of aggregate evaluation (i.e. summations and subtractions only) by delaying the multiplication by $\frac{1}{\sqrt{k}}$. The Johnson-Lindenstrauss lemma [22] guarantees that pairwise distances between documents will be approximately preserved in the resulting representation. The efficient nature of this method makes it suitable for large document collections which make other methods too computationally expensive.

3.5 Latent Dirichlet allocation

Latent Dirichlet Allocation (LDA) [23] is a generative probabilistic model which allows transforming text documents (in the form of word occurrence counts) into representations of lower dimension in the so-called *topic space*. The underlying idea in this method is that word occurrences in documents (observations) can be explained by regarding each document as a mixture of underlying topics. In this context, topics are interpreted as probability distributions over words, and documents are interpreted as probability distributions over topics. Of course, those distributions are inferred automatically from a training corpus.

In practice, this algorithm can be used to estimate the probability of words occurring given a specific topic. In addition, a documents might be represented as a mixture of topics (i.e., a point in the *topic space*). This low-dimensional representation of documents is often used for document classification [24] and retrieval [25], and has been applied to generate suitable case representations in the context of textual CBR [26].

4 Proposed method

This section introduces the proposed hybrid Neural-CBR framework and describes its building blocks in detail. The core idea of our proposal is the integration of state-of-the-art techniques from the field of deep learning into the CBR methodology for the generation of a suitable case-representation. In this context, the neural network is not used as a regular classifier, but to generate a suitable case-representation feature space. This allows the successive retrieval and reuse CBR stages to attain a high classification

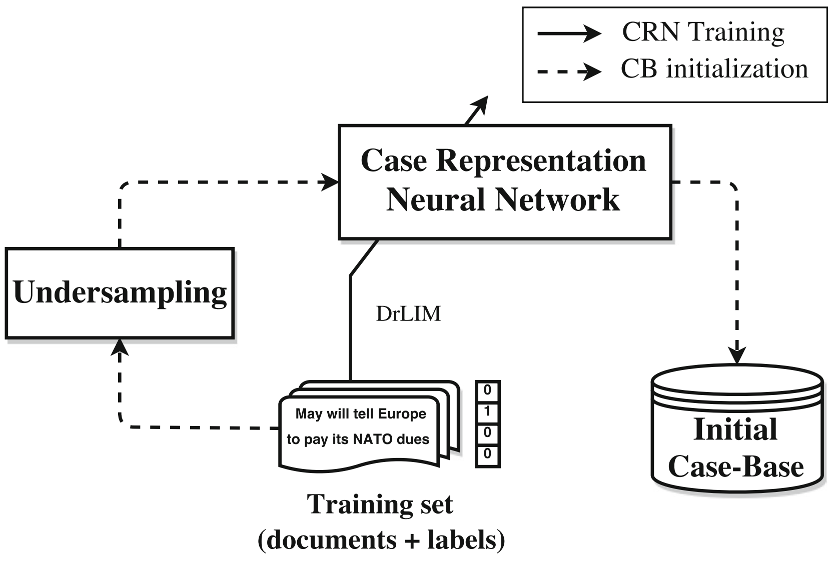

CRN Training
CB initialization
Case Representation Neural Network
Undersampling
DrLIM
May will tell Europe to pay its NATO dues
0
1
0
0
Training set (documents + labels)
Initial Case-Base

Fig. 1 High-level view of the training process of the case-representation network and case-base initialization

accuracy by using a simple distance-based classification approach. Henceforth, we will refer to this neural networks as the Case-Representation Convolutional Network (CR-CNN).

Before the system can be employed to detect clickbait headlines, it must be presented with a training dataset of labeled clickbait/regular headlines. This information is used to train the Case-Representation Network and to populate the Case-Base (CB). As the retrieval stage in CBR is often based on Euclidean distances over cases, the goal for the Case Representation Network is to produce a case representation space where cases of the same category (i.e. clickbait/legitimate headlines) are close to each other. This is achieved by using the DrLIM training method presented in [7]. This process is schematically shown in Fig. 1. More details about the architecture and training process of the CRN can be found in Section 4.1.

Once the Case-Base has been populated, the system is ready to classify new unseen headlines. When an unlabeled headline is presented to the system, it is first forwarded through the CRN to obtain a case representation associated to the headline. The most relevant cases are then retrieved from the Case-Base according to Euclidean distances in the case-representation space. Afterwards, a weighted voting scheme is applied to reuse the retrieved cases and emit a prediction regarding the present case. The revise and reuse stages will enable our system to learn over-time and adapt itself to the specific criteria of particular users. For this reason, the revise stage must be performed by a human who indicates whether a headline has been misclassified by the system. If so, the case representation of the misclassified headline is stored alongside with the correct class label in the Case-Base. In addition, the CRN might be fine-tuned to refine the case representation space. This process is intended to enable the correct classification of future cases which are similar in some sense to previously revised/retained cases. The CBR cycle followed by an incoming case is illustrated in Fig. 2.

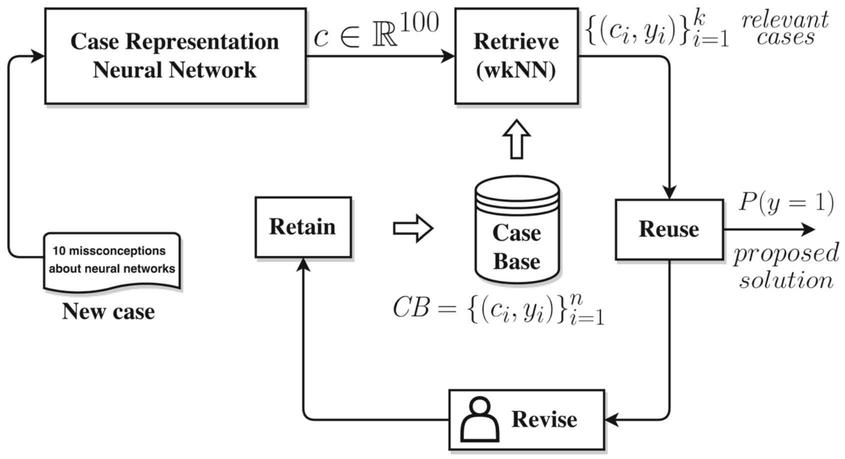


Fig. 2 Overall architecture of the proposed hybrid case-based reasoning system

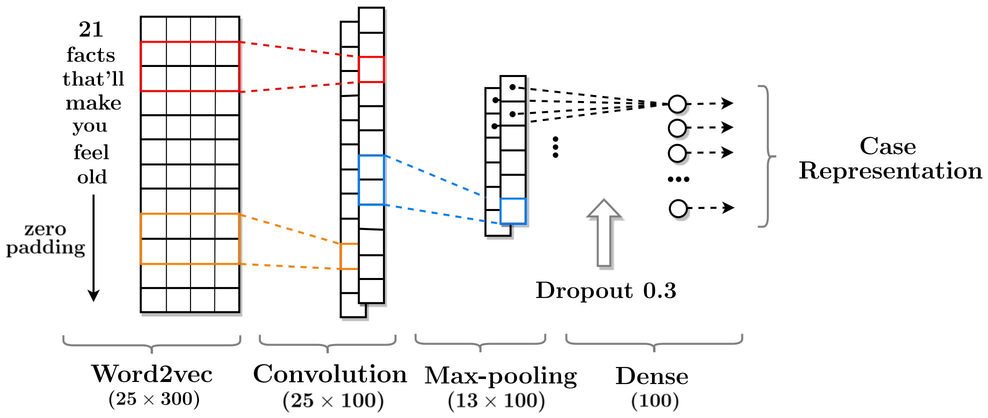


Fig. 3 Overall architecture of the case-representation neural network

4.1 Case representation neural network

The Case Representation Neural Network is responsible for generating meaningful case representations to enable high accuracy rates by the successive CBR stages. However, building such neural network is not a trivial task, as there are several difficulties which must be addressed:

1. The input of the network is in the form of textual information, which must be transformed to fixed-length numerical vectors. In addition, natural language exhibits a huge amount of different patterns. Extracting discriminative information based on those patters can be a challenging problem, especially when a limited amount of training information is available. This difficulty was addressed by applying the so called *transfer learning* approach [27].
2. Natural language has a temporal dimension. Much of the meaning in a sentence is determined by the order in which words occur, rather that the mere set of words employed. All the case representations for textual information presented in Section 3 generate a feature vector invariant to the order of occurrence of words or n-grams in the original documents. In spite of the simplicity of this approach, much information is lost. Ideally, the CRN should take advantage of the information provided by the order of words in sentences.
3. As mentioned before, the CRN will not be used to produce the final prediction concerning a headline, but to generate a suitable case representation to be used in the successive CBR stages. Usually, artificial neural networks are trained by tuning internal weights to minimize the difference between network's output and a desired output. Unfortunately, in this case we do not have a set of desired outputs for training samples. For this reason, a more sophisticated training scheme must be applied, namely the DrLIM method [7].

4.1.1 Network architecture

The overall architecture of the CRN is shown in Fig. 3. The first layer of the network is in charge of generating a fixed-length numerical representation for a given headline. Basically, it consists of a dictionary which assigns a 300-dimensional numerical vector to almost each word in the English language. By doing so, a headline containing w words is transformed into a sequence $x_1, \cdots, x_w$ of 300-dimensional vectors. The sequences in the training set are padded to the length of the longest sequence by adding 300-dimensional vectors of zeros. As opposed to the following layers of the network, this layer needs no training phase. Instead, it was initialized with the weights learned by other network in a different but somehow related task, following the spirit of transfer learning [27]. In particular, the word embeddings[2] used by this layer were provided by the authors of [5]. These word embeddings were originally generated by training a modified version of the Skip-gram model [28] (often referred to as the *word2vec* method). Specifically, the vectors were trained on a subset of the Google News dataset, with approximately 100 billion words.

The second layer of the network is a 1-dimensional convolutional layer [6, 29]. Each convolution neuron in the layer has an associated filter $w \in \mathbb{R}^{l\times 300}$ where l is the desired window length. Formally, let $x_{1:w} = x_1 \oplus x_2 \oplus \cdots \oplus x_w$ be the concatenation of word embeddings for a given headline (where $x_i \in \mathbb{R}^{300}$). For an input window $x_{i:i+l}$, the output of a 1D convolutional neuron with a filter w is computed as follows:

$$c_i = f(w \cdot x_{i:i+l-1} + b) \tag{7}$$

[2]The pre-trained word embeddings used in the CRN are publicly available at https://code.google.com/archive/p/word2vec/.

where $f(\cdot)$ is a nonlinear activation function (we used the rectified linear unit [30]) and b is the bias term of the neuron. This filter is applied to every possible window (depending on the padding/stride scheme) to produce a feature map. In our case, this layer used a stride of one, zero-padding,[3] and a total of one hundred convolutional neurons. Hence, the output of this layer is a $w \times 100$ matrix. The third layer consist of a 1-dimensional Max-pooling. This layer is intended to reduce the dimension of feature maps, thus reducing the computational cost and the number of parameters to be learned in successive layers. In particular, our network uses a pool size of two, thus reducing the output of the previous layer from $\mathbb{R}^{w\times 100}$ to $\mathbb{R}^{w/2\times 100}$.

Finally, the fourth layer of the network is a densely connected neuron layer. The neurons in this layer use a hyperbolic tangent activation. The output activation values of these neurons conform the case representation corresponding to a given input headline. Therefore, the size of this layer determines the dimensionality of the case representation. In practice, we found that a dimension of one hundred components is enough to obtain very high accuracy rates.

In addition, a 0.3 dropout[4] was used at the output of the third layer of the Case-Representation Network [31]. We found that this enabled a significant improvement in the accuracy of the system by effectively preventing overfitting.

4.1.2 Network training

Our goal is to train the above described neural network to produce meaningful case representations. In this context, we do not have a predefined desired output for each training sample. For this reason, we are unable to apply the classical neural network training scheme where the error of the network is computed as the difference between the desired output and the actual output. Instead, we applied the DrLIM training method proposed in [7, 32].

In DrLIM, a parametrized function (the CRN in this case) is optimized in such a way that samples from the same class are pulled closer in the output representation and samples from different classes are pushed away. To achieve this, the loss function runs over pairs of samples rather than individual training samples. Formally, given a training set $\{(x_i, y_i)\}_{i=0}^{n}$, this method first generates a a set of training pairs $\{(x_i, x_j, Y_{ij})\}$ where $Y_{ij} = 0$ if x_i and x_j belong to the same class, and $Y_{ij} = 1$ otherwise. At this point, the

[3]In the context of convolutional neural networks, zero-padding refers to the process of padding the input volume with zeros around the border to preserve the spatial size of the input volume independently of the filter size.

[4]Dropout is a commonly used technique to prevent overfitting. It works by randomly dropping the output of neurons during training, effectively reducing co-adaptation.

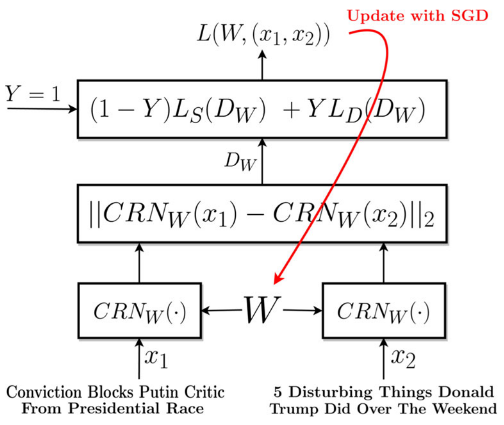

Update with SGD
$L(W,(x_1,x_2))$
$Y=1$
$(1-Y)L_S(D_W) + YL_D(D_W)$
D_W
$||CRN_W(x_1) - CRN_W(x_2)||_2$
$CRN_W(\cdot)$
W
$CRN_W(\cdot)$
x_1
x_2
Conviction Blocks Putin Critic From Presidential Race
5 Disturbing Things Donald Trump Did Over The Weekend

Fig. 4 Siamese architecture used during the training process of the Case-Representation Network by DrLIM

Case-Representation Network is considered as a function $CRN_W(\cdot)$ parametrized by the set W of weights and filters from the different layers of the network. This function is conceptually duplicated to form the so-called *siamese network* architecture. Then, the following computations are performed at each training iteration: (1) a pair of samples x_1, x_2 is forwarded through the siamese networks; (2) the output representations are forwarded to a distance module that computes the Euclidean distance D_W between them; (3) a loss-module calculates the contrastive loss function $L(W, (x_1, x_2))$, which penalizes large D_W values when the pair of samples belongs to the same class and small D_W values when the samples have different classes; and (4) the set of weights W is updated by using a variant of mini-batch stochastic gradient descent[5] (SGD) to minimize the loss function computed in the previous step. Figure 4 illustrates the above described training process by showing a training iteration of the CRN over a pair of samples from different classes; additionally, the two different functions used by the loss module to compute the contrastive loss are displayed in Fig. 5.

After a number of training epochs have been completed, one can expect the Case-Representation Network to generate a case-representation space where samples of the same class exhibit low pairwise distances, while the distances between the representations of headlines from different classes (i.e., clickbait/legitimate) are significantly higher. This will enable a high accuracy rate of the successive CBR stages, which will be in charge of emitting class predictions for incoming test cases. Figure 6 aims to provide insight into the structure of the case-representation space generated by the proposed method. To this extent, it shows the 2-dimensional MDS [34] embedding of some random test cases from our experiments (see Section 5 for

[5]In particular, we used the RMSprop algorithm as implemented in the high-level neural networks library Keras (version 1.2.2) [33].

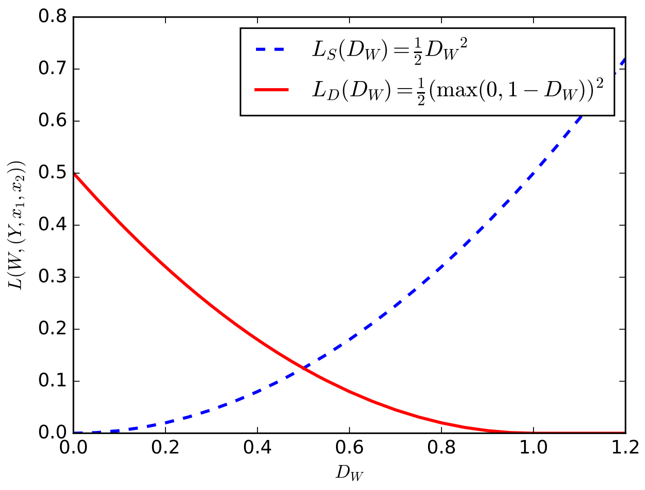


Fig. 5 Auxiliary functions used to compute the contrastive loss. $L_S(\cdot)$ is applied for cases form the same class ($Y = 0$). $L_D(\cdot)$ is used for case pairs with different classes ($Y = 1$) [7]

more details). Interestingly, borderline headlines (i.e., those difficult to classify even for a human being) tend to appear in the transition zone between the clickbait cluster and legitimate cluster.

4.2 Retrieval and reuse stage

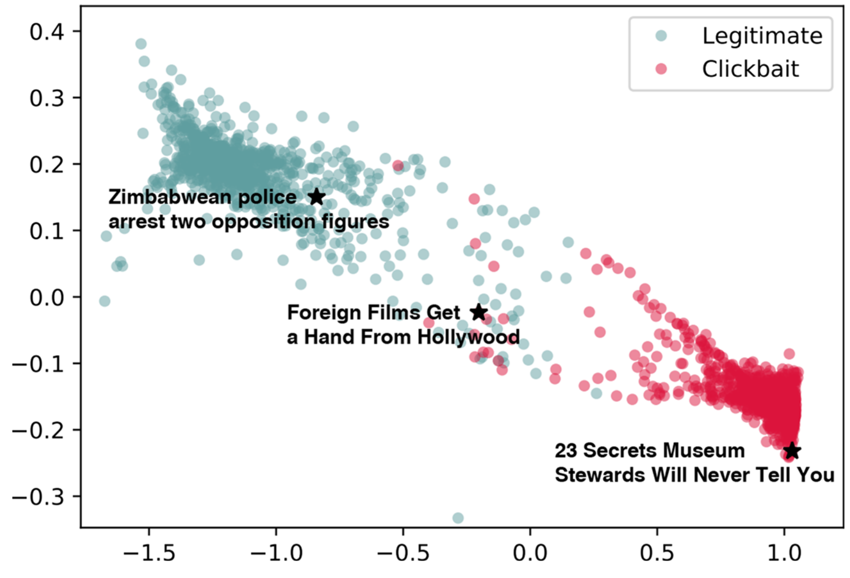
The previous sections described the Case-Representation Network and its training procedure. Once its training process has been completed, the CRN can be used to generate case representations and populate the Case-Base. Formally, if a set $\{(x_i, y_i)\}_{i=0}^n$ of n headlines and their labels is available at the moment of initializing the CB, all these headlines are forwarded through the CRN, yielding n case representations $c_1, \cdots, c_n$, with $c_i \in \mathbb{R}^{100}$. The Case-Base is then populated with all these cases and their corresponding class labels. Formally, $CB = \{(c_i, y_i)\}_{i=0}^n$. However, one of the major limitations of the CBR methodology is the query time, which grows lineally with the size of the Case-Base. To mitigate this problem, we explored the application of different Case-Base reduction methods, namely Nearmiss [35], Condensed Nearest Neighbors (as implemented in [36]) and a naive undersampling of the Case-Base (i.e., dropping at random a fixed percentage of the cases from the Case-Base). Surprisingly, retaining as few as 2,000 cases with the naive undersampling approach preserved almost completely the accuracy obtained with the full Case-Base (see Section 5). This can be partially explained by the invariance to *synonymy* granted by the use of *word2vec* features, where words with similar meanings are mapped closely on the feature space.

After the Case-Base has been initialized, it can be used to differentiate between clickbaits and legitimate headlines. In particular, when a new headline x is presented to the system, it is first processed by the CRN to produce a case representation $c \in \mathbb{R}^{10}$. After this, the k most similar cases (in terms of Euclidean distance) are retrieved from *CB*. These retrieved cases are used to compute the unnormalized weight vector

$$w = \left[\frac{1}{d(c, c_j)} \right]_{j=1}^k \tag{8}$$

where $d(c, c_j)$ is the Euclidean distance between the new case c and its j−th most similar case in *CB*. Then, analogously to the weighted k-Nearest Neighbors algorithm [37], the probability of case c corresponding to a clickbait

Fig. 6 2D MDS embedding of the case-representation for 2,000 headlines sampled at random form our test set (see Section 5 for details). Some representative headlines are also displayed (⋆)



headline[6] is estimated according to the standardized added weights:

$$P(y = 1 \mid c, CB) = \frac{\sum_{j=1}^{k} w_j \cdot I(y_j = 1)}{\sum_{j=1}^{k} w_j} \tag{9}$$

where $I(\cdot)$ is the indicator function which outputs a value of one when it receives a true statement and a value of zero otherwise.

4.3 Revision and retain

Revision and retain stages are the ones that enable the over-time learning capabilities of the CBR methodology. In the context of the proposed CBR system, the revision is carried out by a human user, who determines whether a specific headline has been correctly classified as clickbait or legitimate according to their personal criteria. In addition to a regular retain phase, where misclassified cases are incorporated to the Case-Base, we propose fine-tuning the Case-Representation Network after a number of revised cases have been incorporated into the Case-Base. To this extent, a counter variable r is used to track the number of retained cases since the last fine-tuning of the CRN.

In particular, if the user determines that the system emitted an incorrect class label y for a given headline x, its case representation $c = CRN(x)$ is incorporated into the Case-Base alongside with the correct class label, and the counter variable is increased in one unit:

$$\begin{aligned} CB &\leftarrow CB \cup \ (c, 1 - y) \\ r &\leftarrow r + 1 \end{aligned} \tag{10}$$

When the counter r reaches a predefined threshold T_r, the Case-Representation Network is fine-tuned by executing a small number of iterations of the previously defined training procedure over the samples present in the Case-Base at that moment.[7] After this, the representation of cases in the CB is updated according to the new weight configuration of the CRN. Conveniently, we found that fine-tuning the weights of the final dense layer is enough to enable the adaptation, so the convolutional filters of the first layer of the CRN need not be fine-tuned. Once this fine-tuning process has finished, the counter r is set to zero.

By doing this, the system is able to learn from its mistakes, adapting itself to match the criteria of the user. Section 5 presents experimental results evidencing the over-time learning capabilities of the proposed system, and its ability to adapt to the subjective criteria of users.

5 Experimental results

In this section, we report on a number of experiments confirming the performance of the proposed hybrid CBR framework in the task of clickbait detection. On the one hand, we compare the proposed CBR system with alternative approaches from the literature and classical textual-CBR techniques. On the other hand, the ability of the proposed system to learn over-time and adapt to the subjective criteria of users is also evaluated.

To contribute to the standardization of a public dataset and common evaluation protocol for clickbait detection, we decided to perform our experiments in the largest publicly available dataset, following the evaluation protocol proposed by the original authors. To the best of our knowledge, the biggest publicly available dataset for clickbait detection is the one presented in [1]. This dataset, contains a total of 32,000 headlines from digital newspapers as *BuzzFeed*, *Upworthy*, *New York Times* and *The Guardian* among others. Each headline was manually annotated by at least three human experts, yielding an inter-annotator agreement with Fleiss' $\kappa = 0.79$. The dataset, along with precomputed headline representations generated with the method proposed in [1], is publicly available at https://github.com/bhargaviparanjape/clickbait/.

5.1 Comparison with other approaches

For the first set of experiments, we adopted the evaluation protocol used by the authors of the dataset. In particular, each method was evaluated using a 10-fold cross-validation scheme over the whole dataset. In this first set of experiments, the revision/retain phases of the methods using a CBR approach are not enabled for fair comparison with the non-CBR approach of [1]. In other words, the CBR-based methods are initially provided with the same training samples as the other competing methods. These training samples are then used to populate de Case-Bases using the corresponding case-representations of each approach. Then, during the evaluation/test phase, the different systems are presented with new samples which they have to classify. However, they are not provided with the correct class label after being evaluated on one test sample, so the Case-Bases do not grow during the test phase (similarly, the SVM of [1] is not retrained after predicting the class of each test

[6]We assigned the class label “1” to clickbait headlines and the class label “0” to legitimate headlines.

[7]In practice, we found that assigning revised clickbaits and revised legitimate headlines different class labels than training clickbaits and training legitimate headlines produced a better fine-tunning of the CRN, thus yielding a better test classification accuracy.

sample). The over-time learning capabilities of the CBR-based approaches, granted by the revision/retain stages, are evaluated in the next section.

For hyper-parameter selection, we used nested cross-validation with a grid search over typical values. With this strategy, an inner cross-validation with grid search is used to tune the hyper-parameters and select the best performing values. The outer cross-validation is used to obtain an unbiased estimation of the accuracy of the model's hyper-parameters selected by the inner cross-validation loop [38]. By so doing, we ensure that our accuracy measures will not be biased by the hyper-parameter selection process.

To evaluate the effectiveness of transferring the word embedding representations of *word2vec* [5] when building the CRN, we also evaluated an instance of our method without this technique. Specifically, when not using the *word2vec* embeddings, the architecture of our CRN is the same but the word embeddings of the first layer are initialized at random and optimized by SGD as the rest of the layers in the network.[8] Additionally, we also evaluated the performance of the proposed method (both with *word2vec* embeddings and without them) when applying Case-Base reduction by simple undersampling, as explained in Section 4.2.

Table 1 reports on the average value of several classification evaluation metrics (namely the accuracy, precision, recall and area under the ROC curve) over the ten outer cross-validation folds and the most frequently selected hyper-parameter values for the compared methods. We use SGD_{epochs}, SGD_{batch} and SGD_{lr} to denote the number of training epochs, the batch size and the learning rate used to train the CRN. kNN_k denotes the number of neighbors considered when retrieving the most similar cases from the case base (retrieval stage). Following the notation used in Section 3, LDA_k, LSI_k and RP_k denote the dimension of the output representation generated with each of these dimensionality reduction methods. Finally, *n-gram* indicates the n-gram size (number of words) used to generate the n-gram count or TF-IDF representations of the headlines. Additionally, we computed the average Receiver Operating Curve (ROC) over the folds for each clickbait detection method. The resulting curves for some representative methods are shown in Fig. 7.

Looking at the results listed in Table 1, we see that the best performing approach was the proposed CBR system with *word2vec* embeddings and without Case-Base reduction. This method achieved a 0.994 average area under the ROC curve and a 99% accuracy over the ten

[8]Note that this significantly increases the number of trainable parameters of the CRN, which increases from 160, 200 to 7, 414, 500 when the embedding layer needs to be trained. Also note that the number of parameters of the embedding layer depends on the number of distinct words in the training dataset.

Table 1 10-fold cross validation results of various clickbait detection methods. AUC indicates the corresponding area under the ROC curve

F. Extraction	Classifier	Acc.	Prec.	Rec.	AUC	Parameters
CRN with *word2vec* embedding No Undersampling	CBR (wkNN)	99.0%	98.7%	99.2%	0.994	SGD_{epochs}= 5, SGD_{batch} = 127, SGD_{lr}=0.001, kNN_k=3
with *word2vec* embedding Undersampling (CB_n = 2,000)	CBR (wkNN)	98.9%	98.6%	99.2%	0.994	SGD_{epochs}= 5, SGD_{batch} = 127, SGD_{lr}=0.001, kNN_k=3
CRN with learned embedding No Undersampling	CBR (wkNN)	97.5%	97.5%	97.6%	0.985	SGD_{epochs}= 5, SGD_{batch} = 127, SGD_{lr}=0.001, kNN_k=3
CRN with learned embedding Undersampling (CB_n = 2,000)	CBR (wkNN)	97.5%	97.3%	97.7%	0.985	SGD_{epochs}= 5, SGD_{batch} = 127, SGD_{lr}=0.001, kNN_k=3
n-gram count	CBR (wkNN)	83.3%	98.5%	67.6%	0.90	*n*-gram = 2, kNN_k = 5 L2 similarity
n-gram count + LSI	CBR (wkNN)	90.2%	94.2%	85.7%	0.93	*n*-gram = 2, LSI_k = 100, kNN_k = 3, cosine similarity
n-gram count + LDA	CBR (wkNN)	90.4%	96.1%	84.4%	0.93	*n*-gram = 2, LDA_k = 200, kNN_k = 3, cosine similarity
n-gram count + RP	CBR (wkNN)	88.1%	96.9%	78.6%	0.92	*n*-gram = 2, RP_k = 200, kNN_k= 3, L2 similarity
TF-IDF	CBR (wkNN)	90.9%	94.0%	87.3%	0.95	*n*-gram = 2, kNN_k= 3, cosine similarity
TF-IDF + LSI	CBR (wkNN)	90.4%	95.1%	85.3%	0.95	*n*-gram = {1,2}, LSI_k = 100, kNN_k= 5 , L2 similarity
TF-IDF + LDA	CBR (wkNN)	74.0%	76.8%	67.7%	0.81	*n*-gram = {1,2}, LDA_k = 300, kNN_k = 10, cosine similarity
TF-IDF + RP	CBR (wkNN)	90.5%	95.9%	84.6%	0.93	*n*-gram = 2, RP_k = 200,
kNN_k= 3, cosine similarity						
Chakraborty et al. [1]	SVM (RBF)	92.9%	95.2%	90.4%	0.97	C=1

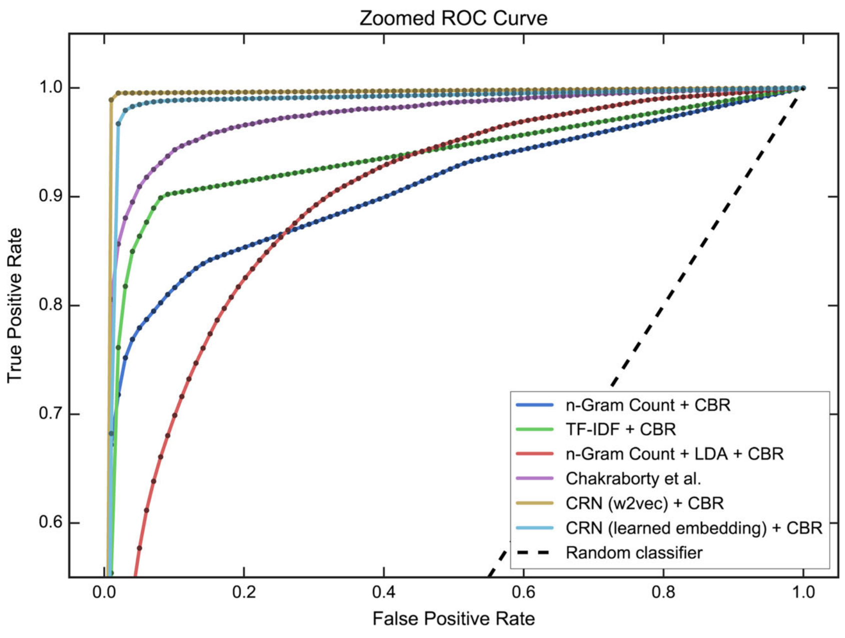


Fig. 7 Zoomed ROC Curves for some representative methods

cross-validation folds. Interestingly, the same system with a random under-sampling for Case-Base Reduction achieved a very similar accuracy. In particular, when all but 2,000 cases were dropped at random from the Case-Base, the system only experienced a 0.1% decrease in its accuracy, which is not statistically significant. This evidences that, once the CRN has been trained, the proposed CBR system can operate with a very small Case-Base and a minimal loss of accuracy. This is highly convenient since the query time of CBR systems (which rely on nearest-neighbor search) directly depends on the size of the Case-Base. The ability of our method to operate with a reduced number of cases in the Case-Base and still maintain a high classification accuracy can be partially explained by the invariance to *synonymy* obtained by using the *word2vec* embeddings at the first layer of the CRN. Thanks to the meaningful representation of words provided by *word2vec*, phrases containing different but semantically related words are mapped close together in the case-representation space, even if those words were not present in the headlines of the training dataset. As a consequence, it is not necessary to store a large amount of samples to fully characterize the clickbait/non-clickbait classes. To illustrate this idea, Fig. 8 shows the activations at the output of the convolution layer of a CRN with the embedding layer trained from scratch and a CRN with the embedding layer transfered from *word2vec* for

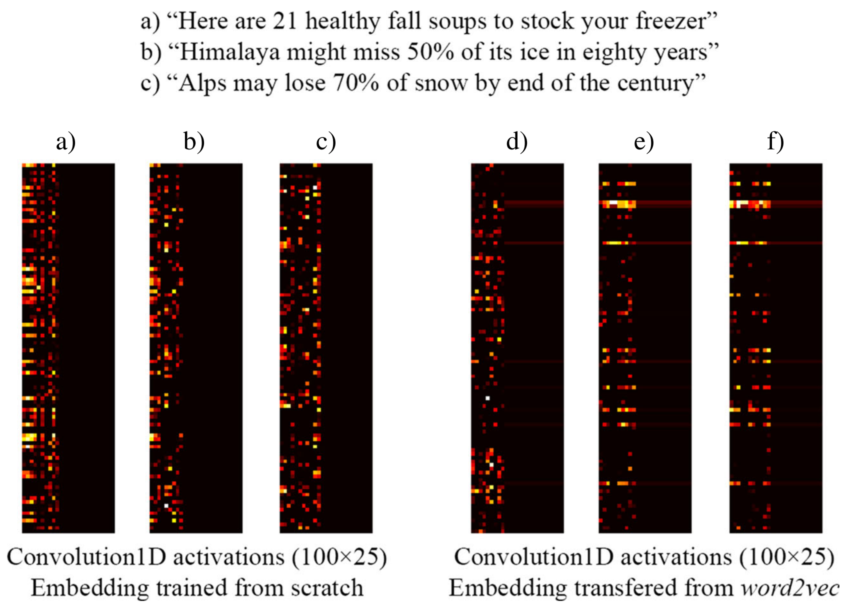


Fig. 8 Activations at the output of the convolution layer of a CRN with the embedding layer trained from scratch (left) and a CRN with the embedding layer transfered from *word2vec* [5] (right) for some sample headlines

three different headlines. As we can see, even though the meaning of headlines b) and c) is quite similar, they use a completely different set of words to express it. Given this semantical similarity, the representation generated for these headlines by the CRN should be approximatively equal, so it would be enough to store one of them in the Case-Base to correctly classify similar incoming headlines. As we can see in the figure, the CRN with *word2vec* embeddings exhibits this desired invariance to semantically similar words, so the activations associated to headlines b) and c) are similar and different from those of headline a), which is clearly a clickbait. Conversely, the CRN with trained word embeddings fails to identify the semantical similarity of headlines b) and c).

The second best performance was achieved by a SVM trained on the features proposed by Chakraborty et al., which achieved a 0.97 average AUC. As we can see, a significant gap exists between the accuracy of the proposed system and that of Chakraborty et al.. The success of the proposed system in the task of clickbait detection is probably due to the use of modern text representation techniques such as *word2vec* and the metric learning approach, as opposed to the hand-crafted representation used in [1]. Note that direct comparison with other clickbait detection methods reviewed in Section 2 is not possible due to the use of additional information (e.g. linked content [4] or mentions and hashtags [3]) which is not available in our target application scenario. For their part, classical text-representation methods traditionally used in the context of textual CBR (e.g., LSI, LDA, RP and TF-IDF) performed poorly in the task of clickbait detection. Among them, the best performing option was TF-IDF with the cosine-similarity, which achieved a 0.95 average AUC.

This poor performance is probably due to the limitations of classical representations used for textual CBR. In particular, these representations do not account for the ordering of words in the documents, so a valuable source of discriminative information is lost. Also, these representations are constructed based on the frequency of words in the training dataset, so words with similar meanings which do not appear in the training set are neglected. As a consequence, using a larger training dataset would probably increase the accuracy of these approaches, but of course this is not always possible due to time and cost constraints.

5.2 Adaptivity simulation

The definition of clickbait is not formal and might be perceived differently by different users. Moreover, even if a specific user agrees on the "standard" notion of clickbait, he might still be interested in reading clickbait headlines related to a specific topic. For this reason, automatic clickbait detection systems should be capable of adapting and modifying their classification criteria to match the sensibility of a specific user. This section reports on a series of experiments performed to support our claim that the proposed CBR system is capable of such adaptation. To this extent, we simulate a number of revision/retain stages by hypothetical users with criteria that are different to that of the annotators of the original dataset.

In our experiments, the labels provided in the original dataset [1] are considered to reflect the standard or normative notion of clickbait, rather than the subjective likes of one specific user (recall that the labels in this dataset were assigned by majority voting of at least three human experts, with an agreement of $\kappa = 0.79$). In this regard, the different models for clickbait detection compared in this section are always initially trained with these original labels. As a consequence, we expect those models to initially adhere to the standard notion of what clickbaits are. As explained before, a desired property for a clickbait-blocking system is the ability to adapt itself to the subjective criteria of final users. For instance, if the user consistently indicates that clickbait headlines of a specific topic should not be blocked, the system should learn that the headlines related to that topic are not perceived as clickbaits by the user, even if they clearly contain clickbait patterns or language. Analogously, the user might perceive headlines about certain topics as clickbaits, even if those are not clickbaits according to the standard notion, so an ideal clickbait detection system should adapt itself to meet these user-preferences. At the same time, the system should preserve its standard behavior for headlines related to topics for which the user has not manifested a discrepancy with the standard criteria.

To evaluate the ability of the different clickbait detection methods compared in this paper to perform this adaptation, we created two new datasets based in the original dataset by Chakraborty et al. [1]. These datasets where generated by arranging a 90-10% train/test split on the original dataset. Then, some of the labels in the test subset where modified to simulate the subjective criteria of two hypothetical users. By so doing, the clickbait detection models are initially trained on the original/standard labels and evaluated against the modified/subjective labels.

In particular, we considered the following two hypothetical users to create the datasets: on the one hand, the first user has a strong interest in the world of costumes. For this reason, he is willing to read every headline related to this topic, even if it was considered as a clickbait by the original annotators. On the other hand, the second user is tired of reading news about US politics, and thus displays no interest at all in headlines on this topic. In other words, he wants the system to block headlines reporting on US politics, even if they do not fit in the standard definition of clickbait. As a consequence, the datasets were generated as follows:

Table 2 Experimental results on CBR learning (Halloween dataset)

	Retain Phase Disabled				Retain Phase Enabled			
Case Representation	Acc.	Prec.	Rec.	AUC	Acc.	Prec.	Rec.	AUC
CRN with *word2vec* embedding	91.4%	84.7%	99.3%	0.923	97.3%	95.4%	99.0%	0.985
CRN with learned embedding	88.2%	80.6%	97.6%	0.909	91.4%	89.6%	91.7%	0.954
Word n-gram count	80.4%	86.1%	68.4%	0.858	80.1%	85.2%	68.6%	0.865
Word n-gram count + LSI	69.7%	61.4%	91.5%	0.859	69.6%	61.5%	90.6%	0.858
Word n-gram count + LDA	81.9%	77.7%	84.9%	0.885	81.9%	77.9%	84.7%	0.888
Word n-gram count + RP	71.1%	62.9%	90.4%	0.731	71.3%	63.2%	90.2%	0.733
TF-IDF	85.3%	83.1%	85.4%	0.881	85.4%	83.4%	85.1%	0.885
TF-IDF + LSI	70.6%	61.9%	93.7%	0.869	70.9%	62.3%	93.4%	0.871
TF-IDF + LDA	72.7%	71.0%	68.6%	0.801	72.8%	71.2%	68.5%	0.802
TF-IDF + RP	86.2%	83.9%	86.6%	0.887	86.3%	84.2%	86.4%	0.892

1. Halloween dataset: this dataset was created by arranging a 90-10% train/test split on the original dataset. All the headlines related to the topic “costumes” (i.e., those containing the words *Halloween*, *Costume* or *Costumes*) were manually re-labeled as non-clickbaits or legitimate headlines. As mentioned before, all the re-labeled headlines were placed in the test set to ensure that clickbait detection systems can only learn about the subjective criteria of the first hypothetical user during test time.
2. US politics dataset: this dataset was also generated by arranging a 90-10% train/test split on the original dataset. However, in this case the headlines related to the topic “US politics” (i.e., those containing the words *Obama*, *Hillary* or *Trump*) where re-labeled as clickbaits. One more time, all the re-labeled headlines were placed in the test set.

In this context, an adaptable clickbait detection system should be able to modify its behavior to match the sensibility of the specific user after a number of revision/retain phases have occurred. For our experiments, we evaluated the different CBR models discussed in this paper on the above described synthetic datasets. To simulate the revise stage, each time a CBR model misclassified a headline, we provided it with the correct class label, so the corresponding revision/retain phases could be executed. To assess the effectiveness of the revise/retain stages, we evaluated the different CBR models with and without the revision/retain phases enabled. The different models were parametrized as in the previous section. Additionally, the threshold for CRN fine-tuning T_r in the proposed model was set to 10 (see Section 4.3). The results of these experiments in the Halloween dataset are listed in Table 2. The corresponding results for the US politics dataset are shown in Table 3.

Table 3 Experimental results on CBR learning (USA-Politics dataset)

	Retain Phase Disabled				Retain Phase Enabled			
Case Representation	Acc.	Prec.	Rec.	AUC	Acc.	Prec.	Rec.	AUC
CRN with *word2vec* embedding	90.4%	99.3 %	83.4 %	0.918	96.0%	98.1%	94.7%	0.979
CRN with learned embedding	88.9%	98.1 %	81.8 %	0.911	89.8%	92.1%	89.4%	0.950
Word n-gram count	75.0%	98.6%	56.0%	0.840	76.0%	98.2%	58.0%	0.852
Word n-gram count + LSI	82.1%	94.2%	72.4%	0.867	82.2%	94.3%	72.5%	0.870
Word n-gram count + LDA	80.5%	87.4%	76.1%	0.872	80.6%	87.4%	76.3%	0.872
Word n-gram count + RP	74.5%	73.4%	85.3%	0.861	74.7%	73.4%	85.6%	0.864
TF-IDF	82.7%	95.7%	72.2%	0.871	82.9%	95.6%	72.7%	0.876
TF-IDF + LSI	88.8%	93.8%	83.1%	0.940	89.1%	94.1%	83.6%	0.944
TF-IDF + LDA	72.8%	82.9%	64.6%	0.810	72.7%	82.4%	65.0%	0.809
TF-IDF + RP	89.5%	96.1%	82.6%	0.925	89.4%	96.1%	82.4%	0.926

As expected, in the case of the Halloween dataset most methods exhibit a high recall with low precision scores when the retain phase is disabled. Since the retain phase is disabled, the different methods cannot learn the subjective criteria of the first simulated user, who considers clickbaits related to a certain topic as legitimate headlines. Therefore, those legitimate headlines (as perceived by the simulated user) are misclassified as clickbaits, resulting in low precession results. An analogous reasoning explains the low recall scores achieved by the different methods when evaluated on the USA-Politics dataset without retaining enabled.

When the retain phase is enabled, the different methods progressively incorporate revised cases into their Case-Bases. In the traditional approaches, this usually leads to a adjustment in the Prec/Rec balance, but little or no improvement in the classification accuracy and AUC. In other words, due to the simplicity of their case representation, traditional models fail to exploit the information about the subjective criteria of users provided in the form of revised cases.

Conversely, the proposed system was able to adapt itself to the specific criteria of our simulated users, increasing its accuracy when feedback from the user (in the form of revised cases) was available. For instance, the accuracies of the proposed system (with *word2vec* embeddings) on the Halloween dataset with and without the retain phase enabled were 97.3% and 91.4% respectively. Similarly, the accuracy of the proposed model when evaluated on the USA-Politics dataset went from 90.4% to 96.0% when the retain phase was enabled. Regarding the use of a CRN with embeddings learned from scratch (instead of transferring them from *word2vec*), this approach displayed little over-time learning capabilities, and a significantly lower accuracy.

6 Conclusions and future work

This paper has presented and evaluated a novel hybrid Neural-CBR system for adaptable clickbait detection. The experimental results presented in this article evidence that the proposed approach outperforms both classic textual-CBR document representations and non-CBR clickbait detection methods present in the literature. In addition, we presented experimental results supporting our claim that the proposed system is able to learn over-time and adapt itself to the specific criteria of different users.

To the best of our knowledge, this is the first time the CBR methodology has been successfully combined with *wor2vec* [28] word-embeddings and deep metric learning techniques (i.e., DrLIM [7, 39]). Using metric learning to define the case representation in the CBR system allowed us to generate a suitable case representation for high-accuracy distance-based classification. In addition, we were able to greatly reduce the size of the Case-Base without a significant accuracy loss by using an inexpensive undersampling method. This was possible thanks to the quality of the case representation, where cases are highly clustered according to their class label. Nevertheless, this case representation showed a great versatility, allowing an effective adaptation to different users' criteria.

In the future, we plan to further investigate the accuracy and adaptability of the proposed method in a real-world evaluation setup. In addition, given the success of our approach in the task of clickbait detection, we believe that the proposed framework can be applied in different application domains. For instance, our method could be applied in the context of clinical record retrieval [26], creating a case representation space where records of patients with similar diseases would exhibit a high similarity.

Acknowledgments The research of Daniel López-Sánchez has been financed by the Ministry of Education, Culture and Sports of the Spanish Government (University Faculty Training (FPU) programme, reference number FPU15/02339). In addition, this work has been supported by project MOVIURBAN: Máquina social para la gestón sostenible de ciudades inteligentes: movilidad urbana, datos abiertos, sensores móviles. SA070U 16. Project co-financed with Junta Castilla y León,Consejería de Educación and FEDER funds.

References

1. Chakraborty A, Paranjape B, Kakarla S, Ganguly N (2016) Stop clickbait: detecting and preventing clickbaits in online news media. In: 2016 IEEE/ACM International conference on advances in social networks analysis and mining (ASONAM). IEEE, pp 9–16
2. Chen Y, Conroy NJ, Rubin VL (2015) Misleading online content: recognizing clickbait as false news. In: Proceedings of the 2015 ACM on workshop on multimodal deception detection. ACM, pp 15–19
3. Potthast M, Köpsel S, Stein B, Hagen M (2016) Clickbait detection. In: European Conference On Information Retrieval. Springer, pp 810–817
4. Biyani P, Tsioutsiouliklis K, Blackmer J (2016) 8 amazing secrets for getting more clicks: detecting clickbaits in news streams using article informality. In: Proceedings of the Thirtieth AAAI conference on artificial intelligence. AAAI Press, pp 94–100
5. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119
6. Kim Y Convolutional neural networks for sentence classification, arXiv:1408.5882
7. Hadsell R, Chopra S, LeCun Y (2006) Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE computer society conference on computer vision and pattern recognition, vol 2. IEEE, pp 1735–1742
8. Spirin N, Han J (2012) Survey on web spam detection: principles and algorithms. ACM SIGKDD Explor Newslett 13(2):50–64

9. Abbasi A, Zhang Z, Zimbra D, Chen H, Nunamaker JF Jr (2010) Detecting fake websites: the contribution of statistical learning theory. Mis Quart 435–461
10. Gianotto A Downworthy: a browser plugin to turn hyperbolic viral headlines into what they really mean, downworthy. snipe. net.
11. Fürnkranz J (1998) A study using n-gram features for text categorization. Austrian Res Inst Artif Intell 3(1998):1–10
12. Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63(1):3–42
13. Friedman JH (2002) Stochastic gradient boosting. Comput Stat Data Anal 38(4):367–378
14. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
15. Socher R, Perelygin A, Wu JY, Chuang J, Manning CD, Ng AY, Potts C et al (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the conference on empirical methods in natural language processing (EMNLP), vol 1631. Citeseer, p 1642
16. Richter MM, Weber R (2013) Case-based reasoning: a textbook. Springer Science & Business Media
17. Fan R-E, Chang K-W, Hsieh C-J, Wang X-R, Lin C-J (2008) LIBLINEAR: a library for large linear classification. J Mach Learn Res 9:1871–1874
18. Baeza-Yates R, Ribeiro-Neto B et al (1999) Modern information retrieval, vol 463. ACM Press, New York
19. Manning CD, Raghavan P, Schütze H et al (2008) Introduction to information retrieval, vol 1. Cambridge University Press, Cambridge
20. Achlioptas D (2001) Database-friendly random projections. In: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems. ACM, pp 274–281
21. Rehurek R, Sojka P (2010) Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks. Citeseer
22. Dasgupta S, Gupta A (2003) An elementary proof of a theorem of johnson and lindenstrauss. Random Struct Algor 22(1):60–65
23. Hoffman M, Bach FR, Blei DM (2010) Online learning for latent dirichlet allocation. In: Advances in neural information processing systems, pp 856–864
24. Li K, Xie J, Sun X, Ma Y, Bai H (2011) Multi-class text categorization based on lda and svm. Procedia Eng 15:1963–1967
25. Wei X, Croft WB (2006) Lda-based document models for ad-hoc retrieval. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, pp 178–185
26. Arnold CW, El-Saden SM, Bui AA, Taira R (2010) Clinical case-based retrieval using latent topic analysis. In: AMIA annual symposium proceedings, vol 2010. American Medical Informatics Association, p 26
27. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Know Data Eng 22(10):1345–1359
28. Mikolov T, Chen K, Corrado G, Dean J Efficient estimation of word representations in vector space, arXiv:1301.3781
29. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. J Mach Learn Res 12:2493–2537
30. Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10), pp 807–814
31. Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
32. Chopra S, Hadsell R, LeCun Y (2005) Learning a similarity metric discriminatively, with application to face verification. In: IEEE Computer society conference on computer vision and pattern recognition, 2005. CVPR 2005, vol 1. IEEE, pp 539–546
33. Chollet F (2015) Keras, https://github.com/fchollet/keras
34. Borg I, Groenen PJ (2005) Modern multidimensional scaling: theory and applications. Springer Science & Business Media
35. Mani I, Zhang I (2003) knn approach to unbalanced data distributions: a case study involving information extraction. In: Proceedings of workshop on learning from imbalanced datasets
36. Lemaître G, Nogueira F, Aridas CK Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning, CoRR arXiv:1609.06570
37. Hechenbichler K, Schliep K Weighted k-nearest-neighbor techniques and ordinal classification
38. Cawley GC, Talbot NL (2010) On over-fitting in model selection and subsequent selection bias in performance evaluation. J Mach Learn Res 11:2079–2107
39. Sun Y, Chen Y, Wang X, Tang X (2014) Deep learning face representation by joint identification-verification. In: Advances in neural information processing systems, pp 1988–1996

Daniel López-Sánchez obtained a Computer Science degree (best GPA award) in 2015 and an MSc in Artificial Intelligence in 2016 at the University of Salamanca (Spain). In 2016, he received the FPU grant from the Spanish Government and started pursuing a PhD at the BISITE Research Group. His research interest focuses in the field of machine learning, especially in the subfields of dimensionality reduction and kernel methods. He is also interested in developing novel applications of the Deep Learning paradigm. He has participated as a co-author in papers published in recognized international conferences and journals.

Jorge Revuelta Herrero born and growth in Salamanca, Spain. Obtained a Computer Science degree in 2015 and MSc in Artificial Intelligence in 2016 at the University of Salamanca (Spain). In 2017, he received the research internship by the Junta de Castilla y León and started pursuing a PhD at the BISITE Research Group. His research interest focuses in the field of social networks, Big Data, Cloud Computing and data visualization. He also is Senior iOS Developer at Ebikemotion SL. He has participated as a co-author in papers published in recognized international conferences and journals.

Angélica González Arrieta received a PhD in Computer Science from the University of Salamanca in 2000. She is currently a Lecturer in Salamanca's University Department of Computer Science and has attended several Master's courses. She is further a professor and tutor for UNED (Universidad Española de Educación a Distancia, Spain's Open University). In the past, she carried out relevant administrative tasks, such as Academic Secretary of the Science Faculty (1996-2000) and Chief of Staff for the University of Salamanca (2000-2003). From 1990, she has cooperated with the Home Ministry and from 2008 with the Home and Justice Counsel of the local government.

Juan M. Corchado received a PhD in Computer Science from the University of Salamanca in 1998 and a PhD in Artificial Intelligence (AI) from the University of Paisley, Glasgow (UK) in 2000. At present, he is Vice President for Research and Technology Transfer since December 2013 and a Full Professor with Chair at the University of Salamanca. He is the Director of the Science Park of the University of Salamanca and Director of the Doctoral School of the University. He has been elected twice as the Dean of the Faculty of Science at the University of Salamanca. Since January 2015, Juan Manuel Corchado is Visiting Professor at Osaka Institute of Technology. He is also member of the Advisory group on Online Terrorist Propaganda of the European Counter Terrorism Centre (EUROPOL). Juan Manuel is also editor and Editor-in-Chief of Specialized Journals like ADCAIJ (Advances in Distributed Computing and Artificial Intelligence Journal), IJDCA (International Journal of Digital Contents and Applications) and OJCST (Oriental Journal of Computer Science and Technology).