Using Modified Term Frequency to Improve Term Weighting for Text Classification

By,

Bhadra Giri (181CO113)

Nishtha Kumari (181CO236)

Introduction

- ***Text classification*** (TC) is an essential task of ***natural language processing*** (NLP). In order to improve the performance of TC, term weighting is often used to obtain effective text representation by assigning appropriate weights to each term.
- A term weighting scheme is generally composed of:
  - term frequency factor,
  - collection frequency factor
  - normalization factor.
- Here, a new term frequency factor called ***modified term frequency*** (MTF) is proposed.
- Then, a new term weighting scheme by combining MTF with an existing collection frequency factor called ***modified distinguishing feature selector*** (MDFS). The scheme is denoted by ***MTF-MDFS (MDFS-based MTF).***

$$TWS = TermFrequencyFactor \cdot CollectionFrequencyFactor \cdot NormalizationFactor,$$

- The most commonly used term frequency factor is the ***raw term frequency*** (TF), which represents the number of occurrences of terms in a document. Moreover, some variants of TF can also achieve good performance in TC, e.g., logarithm of TF and square root of TF, etc.
- The collection frequency factor states the distribution information of a specific term in the entire corpus, and its value generally reflects the importance of the term for classification.

- The term frequency in each document should be adjusted according to the length information of all training documents, which is different from the cosine normalization to normalize the term weight. Hence, that information may be lost in the normalization process.
- For example, the normalized term weights may no longer be applicable to some base classifiers that require frequency information, such as multinomial naive Bayes (MNB).
- Based on these premises, the length information of all training documents is introduced into the term frequency factor and propose a new term frequency factor called ***modified term frequency*** (MTF).
- Then a new term weighting scheme is proposed by combining MTF with an existing collection frequency factor called ***modified distinguishing feature selector*** (MDFS).

MTF

- For each document ***dk*** (k=1,2,…,n), the density function ***ρ(l)*** of term ***ti*** is defined as:

$$\rho(l) = s \cdot \frac{f_{ik}}{l},$$

where ***s*** is a constant, and ***fik*** is the raw term frequency of term ***ti*** in ***dk***.

To resize the term frequency, we calculate the integral on the same interval [l(dk),l(dk)+avg_l], where ***l(dk)*** is the document length of dk, which is defined as the sum of all term frequencies, and ***avg_l*** is the average document length of all training documents. The resized term frequency (simply denoted as **RTF)** is defined as:

$$RTF(t_i, d_k) = \int_{l(d_k)}^{l(d_k)+avg\_l} \rho(l)\, dl.$$

the ***RTF*** of term ***ti*** in document ***dk*** is

$$\begin{aligned} RTF_2(t_i, d_k) &= \int_{l(d_k)}^{l(d_k)+avg\_l} s \cdot \frac{f_{ik}}{l} dl \\ &= s \cdot f_{ik} \cdot \int_{l(d_k)}^{l(d_k)+avg\_l} \frac{dl}{l} \\ &= s \cdot f_{ik} \cdot \log_e \left(1 + \frac{avg\_l}{l(d_k)}\right). \end{aligned}$$

- Assume that the average length of all training documents is equal to the effective length of the document. When **l(dk)** is equal to ***avg_l, RTF(ti,dk)*** is equal to ***fik***. So in this case, we can figure out that s=1/loge2=log2e.
- $$RTF_2\left(t_i, d_k\right) = \log_2 e \cdot f_{ik} \cdot \log_e\left(1 + \frac{avg\_l}{l(d_k)}\right)$$
  $$= f_{ik} \cdot \log_2\left(1 + \frac{avg\_l}{l(d_k)}\right).$$
- We consider that when the document length differs greatly, it is easy to cause the RTF value to be too large or too small. So we also use a square root function to suppress this effect. Based on these premises, we propose a new term frequency factor called ***modified term frequency*** (MTF), which is defined as:

$$MTF\left(t_i, d_k\right) = \sqrt{f_{ik} \cdot \log_2\left(1 + \frac{avg\_l}{l(d_k)}\right)}.$$

MDFS

- MDFS makes full use of the distribution information of terms in all training documents, and it is more effective than other collection frequency factors for term weighting in most cases.
- MDFS is derived from a widely accepted term selection method called ***distinguishing feature selector*** (DFS). The DFS argues that an ideal term (feature) selection method should assign high scores to distinctive terms and assign low scores to irrelevant terms. Specifically, DFS must meet four requirements:

1. **If a term occurs frequently in a single class and does not occur in other classes, it is distinctive and must be assigned a high score.**
2. **If a term occurs frequently in all classes, it is irrelevant and must be assigned a low score.**
3. **If a term occurs rarely in a single class and does not occur in other classes, it is irrelevant and must be assigned a low score.**
4. **If a term occurs in some of the classes, it is relatively distinctive and must be assigned a medium score.**

- DFS defines the term selection score of term ti as follows:

$$DFS(t_i) = \sum_{j=1}^{q} \frac{P(c_j|t_i)}{P(\bar{t}_i|c_j) + P(t_i|\bar{c}_j) + 1}.$$

- By analyzing the formula of DFS, we found that the specificity of a term in a class has not been adequately demonstrated. To address this issue, we argue that an ideal term selection score should satisfy the fifth requirement:
- ***"For class cj, d(ti,cj) must be large and d($\bar{t}$i,cj) must be small; For class $\bar{c}$j (absence of class cj), d($\bar{t}$i,$\bar{c}$j) must be large and d(ti,$\bar{c}$j) must be small".***
- Then we modified the DFS and proposed a new collection frequency factor simply denoted by MDFS. In MDFS, each term ***ti*** has a class-specific score for each class ***cj.***
- In addition, a weighting factor is used for class-specific scores to further reflect the contributions of terms to a single class. MDFS is defined as a global weighting factor, calculated by the weighted sum across all class-

$$MDFS\left(t_i\right) = \sum_{j=1}^{q} w_{ij} \cdot MDFS_{cs}\left(t_i, c_j\right),$$

- where ***wij*** and ***MDFScs(ti,cj)*** represent the weighting factor and class-specific score of term ***ti*** for class ***cj***, respectively. They are defined respectively as follows:

$$w_{ij} = \log_2\left(1 + \frac{d(t_i,c_j)}{\max(1,d(t_i,\overline{c_j}))} \cdot \frac{d\left(\overline{t_i},\overline{c_j}\right)}{\max\left(1,d\left(\overline{t_i},c_j\right)\right)}\right),$$

$$MDFS_{cs}\left(t_i, c_j\right) = \frac{P(c_j|t_i)P\left(\overline{c}_j|\overline{t}_i\right)}{P\left(\overline{t}_i|c_j\right)+P(t_i|\overline{c}_j)+1}.$$

Finally, by combining MTF with MDFS, we propose a new term weighting scheme called ***MTF-MDFS (MDFS-based MTF).*** The detailed formula is:

$$W_{\text{MTF - MDFS}}\left(t_i, d_k\right) = MTF\left(t_i, d_k\right) \cdot MDFS\left(t_i\right).$$

The extensive comparison results validate the effectiveness of our proposed MTF and MTF-MDFS in terms of the classification performance of widely used base classifiers.

- In terms of ***MNB***, the averaged ***classification accuracy*** of MTF-MDFS on 19 datasets is ***89.60%***, which is much higher than those of LogTF-RFmax (86.95%), SqrtTF-IGM (86.54%) SqrtTF-IGMimp (86.81%) and TF-MDFS (87.82%).
- In terms of ***SVM***, the averaged ***classification accuracy*** of MTF-MDFS on 19 datasets is ***91.46%***, which is also higher than those of LogTF-RFmax (90.94%), SqrtTF-IGM (91.01%) SqrtTF-IGMimp (91.05%) and TF-MDFS (89.73%).

Dataset used:

- 20 Newsgroups: This dataset contains 19997 documents of newsgroup messages, which are divided into 20 classes. Except for 997 documents in one class, there are 1000 documents in each of the remaining 19 classes. To save training time, numbers, punctuation marks and other non-alphabetic characters are removed. At the same time, the letters are converted to lower case.

- MTF-MDFS ALgorithm
- Algorithm 1 MTF-MDFS learning in the training phase (D)
-

  Input: D-a training document set
- Output: All term weights WMTF-MDFS($t\ i$, dk), avg_l, $MDF\ S(t\ i\ )$ ($i = 1, 2, \cdots, m$)

1. for each document dk ($k = 1, 2, \cdots, n$) do
2. Calculate $l(dk\ )$
3. end for
4. Calculate avg_1
5. for each term $t\ i$ ($i = 1, 2, \cdots, m$) and each class cj ($j = 1, 2, \cdots, q$) do
6. Calculate wij by Eq. (11)
7. Calculate $MDFS\ cs(ti\ , cj\ )$
8. end for
9. for each term ti ($i = 1, 2, \cdots, m$) do
10. Calculate $MDFS(t\ i\ )$ by Eq. (10)
11. for each document dk ($k = 1, 2, \cdots, n$) do
12. Calculate $MTF(ti\ , dk\ )$
13. Calculate WMTF-MDFS($ti\ , dk\ )$
14. end for
15. end for
16. return All term weights WMTF-MDFS($t\ i\ , dk\ )$, avg_l, $MDF\ S(t\ i\ )$ ($i = 1, 2, \cdots, m$)

- Algorithm 2 MTF-MDFS learning in the testing phase (d)
-
- Input: d-a test document, avg_l, $MDF\ S(t\ i\ )\ (i = 1, 2, \cdots, m)$
- Output: All term weights WMTF-MDFS($t\ i\ , d$)

1. Calculate $l(d)$
2. for each term $t\ i\ (i = 1, 2, \cdots, m)$ do
3. Calculate $MT\ F(t\ i\ , d)$
4. Calculate WMTF-MDFS($t\ i\ , d$)
5. end for
6. return All term weights

Drawback of the paper

The term frequency factor and the collection frequency factor are equally important for term weighting, both of which are helpful to improve the performance of TC. Currently, they simply combined MTF and MDFS without considering whether they maybe counteract each other to generate inappropriate weights. For example, when the length difference of training documents is particularly large, there is likely to be a situation that the term frequency factor plays a dominant role in term weighting and the role of the collection frequency factor is greatly reduced. Authors believe that the use of more sophisticated combination methods could improve the performance of the current MTF-MDFS and make its advantage stronger. This will be a major direction for our future work. In addition, applying more sophisticated optimization methods to directly search term weights is another interesting topic for future work.

- Accuracy of MTF-MDFS using different models
- Naive Bayes 86%
- SVM 79%
- Logistic Regression 80%
- Xgboost 59%

MTF- ICF

It is proposed by us. We used MTF term frequency factor which is one of the best known till date with ICF collection factor and named it as MTF- ICF.

Accuracy of 89% with naive bayes.

ICF Formula:

log2(1+q/c(ti)

q- the total number of classes.

c(ti)- the number of classes containing term *ti* .

We implemented MTF with other collection frequency also.

- MTF-IDF-ICF: MTF with IDF-ICF
- MTF-RF: MTF with RF
- MTF-PB: MTF with PB
- MTF-TCF-Based: MTF with ICF Based
- TF-MDFS: TF with MDFS
- MTF-DC: MTF with DC

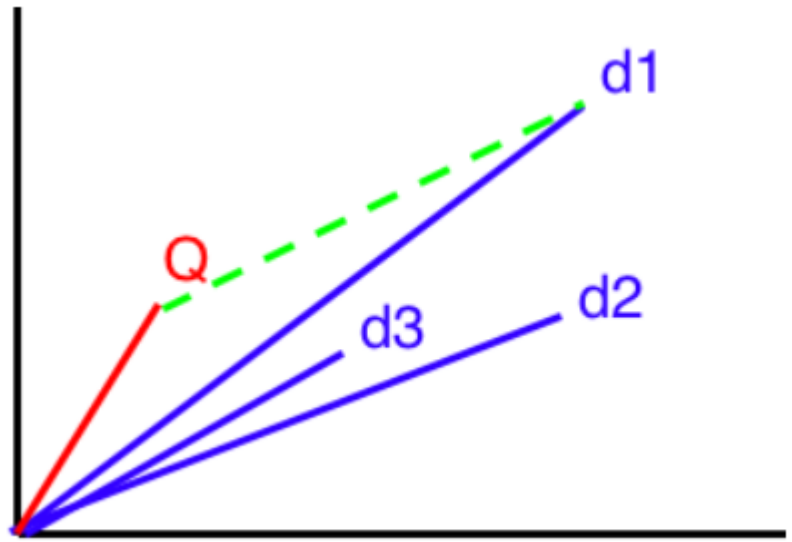
Matching Score and Cosine Similarity

- Matching score is the most simplest way to calculate the similarity, in this method, we **add tf_idf values of the tokens that are in query for every document**. for example, if the query "hello world", we need to check in every document if these words exists and if the word exists, then the tf_idf value is added to the matching score of that particular doc_id. in the end we will sort and take the top k documents.

- **Matching Score** gives relevant documents, but fails when we give long queries and will not be able to rank them properly.

- **Cosine similarity** takes the angle between two non-zero vectors and calculates the **cosine** of that angle, and this value is known as the **similarity** between the two vectors. This **similarity score** ranges from 0 to 1, with 0 being the lowest (the least similar) and 1 being the highest (the most similar).

- Observe the above plot, the blue vectors are the documents and the red vector is the query, as we can clearly see, though the manhattan distance (green line) is very high for document d1, the query is still close to document d1. In these kind of cases cosine similarity would be better as it considers the angle between those two vectors. But Matching Score will return document d3 but that is not very closely related.



Contribution

Proposed two new weighing Scheme by modifying ICF, MDFS and DFS.

MTF-ICF-DFS

- Accuracy with multinomial naive bayes: 88%.
- Formula:

DFS:

$$DFS(t) = \sum_{i=1}^{M} \frac{P(C_i|t)}{P(\bar{t}|C_i) + P(t|\bar{C}_i) + 1},$$

ICF:

$$\log_2 \left(1 + \frac{q}{c(t_i)}\right)$$

New weighing scheme: weight(ti,dk)= mtf*icf*dfs

- MTF-ICF-MDFS
- Accuracy obtained with multinomial naves bayes is 89% (best so far)
- Formula:
-

MDFS:

$$\sum_{j=1}^{q} \log_2 \left(1 + \frac{d(t_i, c_j)}{\max\left(1, d(t_i, \overline{c_j})\right)} \cdot \frac{d(\overline{t_i}, \overline{c_j})}{\max\left(1, d(\overline{t_i}, c_j)\right)} \right) \cdot \frac{P(c_j|t_i) P(\overline{c_j}|\overline{t_i})}{P(\overline{t_i}|c_j) + P(t_i|\overline{c_j}) + 1}$$

ICF:

$$\log_2 \left(1 + \frac{q}{c(t_i)} \right)$$

New weighing scheme: weight(ti,dk) = mtf*icf*mdfs

Some terms:

- fik: the raw term frequency of term $t\ i$ in document dk .
- q: the total number of classes.
- n: the total number of training documents.
- $d(t\ i\ )$ the number of documents containing term $t\ i$.
- $d(t\ i\ ,\ cj\ )$ the number of documents belonging to class cj containing term $t\ i$.
- $d(t\ i\_bar\ ,\ cj)$ the number of documents belonging to class cj not containing term $t\ i$.
- $d(t\ i\ ,\ cj\ \_bar)$ the number of documents not belonging to class cj containing term $t\ i$.
- $d(t\ i\_bar\ ,\ cj\_bar)$ the number of documents not belonging to class cj not containing term $t\ i$
- $P\ (cj\ |t\ i\ )$ the conditional probability of class cj given the presence of term $t\ i$.
- $P\ (cj\ |t\ i\ )$ the conditional probability of the absence of class cj given the absence of term $t\ i$. \
- $P\ (t\ i\ |cj\ )$ the conditional probability of the absence of term $t\ i$ given the presence of class cj .
- $P\ (t\ i\ |cj\ )$ the conditional probability of term $t\ i$ given the absence of class cj .

Contribution

- Mtf-mdfs implemented through the the paper has the best accuracy with Naïve bayes classifier **(86%)**
- As part of our contribution,mtf-icf, mtf-dfs-icf and mtf-mdfs-icf gives **89%**, **88%** and **89%** accuracy respectively.