ISSN 2007-9737

Evaluating Text Summaries using Divergences of the Probability Distribution

Juan Manuel Torres Moreno[1,3], Ligia Quintana Torres[2], Porfirio Toledo Hernández[2]

[1] Université d'Avignon,
Laboratoire Informatique d'Avignon,
France

[2] Universidad Veracruzana,
Facultad de Matemáticas,
Mexico

[3] Polytechnique de Montréal,
Département Génie Informatique et Génie Logiciel,
Canada

juan-manuel.torres@univ-avignon.fr, {liquintana, ptoledo}@uv.mx

Abstract. This paper aims to show that generating and evaluating summaries are two linked but different tasks even when the same Divergence of the Probability Distribution (DPD) is used in both. This result allows the use of DPD functions for evaluating summaries automatically without references and also for generating summaries without falling into inconsistencies.

Keywords. Kullback-Leibler/Jensen-Shannon divergences, probability distribution, natural language processing, automatic text summarization.

1 Introduction

The available number of documents in electronic form has exponentially grown in recent times. It is impossible for a human to absorb this information to the rate that is produced. The multilingualism present in documents on the Internet does not make the problem any easier. One way to deal with the issues raised are the Automatic Text Summarization (ATS) algorithms [32, 33, 17, 16, 34, 24], a Natural Language Processing (NLP) task. ATS is a compression with losing information. A summarization algorithm aims to compress a source document in order to save their essential content. ATS can produce a relevant synthesis of the documents, giving the reader the decision to consult the source literature.

The production of summaries by humans is a highly subjective cognitive process: not only is it necessary to have knowledge of the language in which the document is written but extra-linguistic domain knowledge is also required in order to produce relevant abstracts.

There are different types of summaries depending on: i) the source documents: single-document, multi-document, monolingual, multilingual, generic, guided, etc.; ii) the genre of texts: news, scientific, encyclopedic, etc.; iii) the audience: expert, general public, etc. Other kinds of summarization are also possible: blogs summarization, tweets summarization, etc. For a more complete state of the art, the reader may consult: [17, 16, 24, 34], among other readings.

ATS algorithms, despite not generating quality summaries comparable to those produced by humans, they are capable of generating exploitable summaries that give a reduced view of the source document.

The term “exploitable” means that the automatically generated summary has certain qualities that make it interesting:

1. It has to be shorter than the original (measured in characters, words, sentences, etc.).
2. Good informative content coming from the source document (the informativity).
3. It has to be coherent, grammatically correct and readable (the form of the document).

However, the two latter qualities are highly subjective. How to correctly measure the informativity and the form at the same time? The answer to these questions is still open however partial answers exist [16, 8, 23].

In ATS two processes are clearly differentiated: a) the production of –relevant– summaries, and b) the –correct– evaluation of these summaries. It is not enough to create ATS systems, it is necessary to correctly evaluate the production of such systems. Process (a) presents several intrinsic difficulties: the “comprehension” of the text, the selection of the informative segments, their assembly and their possible rewriting.

The production of summaries can be realized by extraction or abstraction approaches. The extractive approach selects the most informative segments from the source, then the algorithm assembles them into a summary [15, 1, 3]. The abstractive approach aims to generate new and informative texts from the source document [16, 34]. Nowadays, most systems use an extractive approach because the algorithms for extracting sentences have shown technical superiority against complex methods of abstract generation (based on rules or others).

Process (b), quality summary assessment is a task with its own difficulties [24, 16]. In general, summary evaluation can be performed manually (reading and direct evaluation by human judges), automatically (comparing artificial summaries vs. references[1]) or automatically without references (comparing artificial summaries vs. the source document) [34].

[1]References are summaries made by humans (see Appendix C).

The evaluation with references is dominant in the literature because it presents a good correlation with the manual evaluations [11].

In this paper we are interested in automatic evaluation without references. This way of evaluating has a great advantage: not needing human references or judges trying to analyze large volumes of sources and artificial summaries. In addition, this type of evaluation has shown a good correlation with methods using references [14, 34]. However the evaluation without references has incurred severe criticism.

It can be argued that the evaluation algorithm, being able to evaluate summaries, also –potentially– has the capacity to generate them. Supposing that when generating a summary using the weighting function of an evaluation algorithm and measuring the quality of this summary using the same evaluating algorithm is actually cheating. The aim of this paper is to show that, under certain assumptions, using the evaluation function during the generation process does not guarantee that the optimal summary will be obtained at the same moment when it is evaluated.

The rest of the paper is organized as follows: Section §2 discusses the generation and evaluation of summaries. In Section §3, two proofs show the difference between generating and evaluating summaries using divergence of probability distribution. Finally in Section §4 the discussion and conclusions are presented. Three annexes with linguistic motivation complete the paper: A) the representation of documents as probability distribution, B) the automatic generation of summaries and C) their evaluation.

2 Generating and Evaluating Summaries

Through a classical approach, textual documents can be normalized and represented in an appropriate vector space (see A). In this space, there are algorithms that can calculate a relevant summary of the text and other algorithms that evaluate the quality of the informative content of a summary already generated. In this section we will formally define the tasks of generating and evaluating textual summaries.

2.1 Automatically Generation of Summaries

Let $\mathcal{T}$ be a source document consisting of a set of f_i; $i = 1, 2, \ldots, P$ sentences. Let $\mathcal{X} = \{w_1, \ldots, w_j, \ldots, w_n\}$ be the vocabulary[2] of $\mathcal{T}$. The size of $|\mathcal{X}| = n$ words (or tokens). Let f_i be a sentence $\in \mathcal{T}$, consisting of a set of normalized tokens w_j. Each token w_j has a number of occurrences associated with the document $\mathcal{T}$, denoted by $C_w^{\mathcal{T}} \in \mathbb{N}$. The number of occurrences of the token w_j can also be counted in each sentence f_i as $C_w^{f_i}$.

The extractive paradigm of summary generation [15, 1, 3, 16, 34] is currently dominant in literature. This paradigm involves the selection and extraction of the k most relevant (informative) sentences belonging to the source document $\mathcal{T}$. This set of sentences will be denoted by $\mathcal{A}$ and it is calculated through a weighting (or normalized score between $[0, 1]$) of the sentences $f_i \in \mathcal{T}$. The k sentences having higher scores become part of $\mathcal{A}$, the rest are probably deleted.

Formally, let a sentence $f_i = \{w_1, w_2, \ldots\}$ be a sequence of words and a text $\mathcal{T} = \{f_1, f_2, \ldots, f_P\}$ a set of P sentences. Let $\Omega(f_i) \rightarrow [0, 1]$ be a weighting function [3] that measures normalized information (0: null information, 1: maximum information) of a sentence f_i. An automatic summary $\mathcal{A}$ is a subset of k sentences of $\mathcal{T}$, produced by an algorithm. The number k defines the compression rate: $\rho = \frac{k}{P}$. In a summary generated by extraction, the k sentences of $\mathcal{A} \subset \mathcal{T}$ and its vocabulary $\mathcal{Y} \subseteq \mathcal{X}$.

A summarizer is a function G that generates a subset $\mathcal{A}$ having k sentences, $0 < k < P$, such as: $G : \mathcal{T} \rightarrow \mathcal{A}$; $\mathcal{A} = \{f_k\} \in \mathcal{T}$ is an extract of k sentences having the highest $\Omega(f_k)$ weights. k will be set following the compression rate ρ. There are several types of algorithms of sentence extraction (see some heuristics in B).

[2]Set of unique words in a text. In this article a word will be a token separated by whitespace.

[3]The sentence weighting is one of the central problems of ATS. Examples of sentence weighting functions are: the length in words $\Omega(f) = |f|$, the number of keywords (nouns, verbs, ...) of f: $\Omega(f) = \sum_{w \in \text{Keywords}} f$, the sum of the TF IDF of the words: $\Omega(f) = \sum_w TF_w \times IDF_w$ (see A), etc. More complex functions, iterative functions or functions using graphs of sentences are commonly used to weight sentences.

2.2 Automatically Evaluation of Summaries

There are several manual and automatic methods with and without references capable of evaluating text summaries (see C). In this section we only will deal with the automatic evaluations without references. This type of evaluation and its relationship with the summaries generation using Divergence of Probability Distribution constitute the critical point of the paper.

Automatic evaluations without references are based on a measure of divergence between two probability distributions [7, 12, 13, 14]. Two suitable divergence measures are the Divergence, Relative Entropy, or Kullback-Leibler Distance D_{KL} [2] and the Jensen-Shannon Divergence D_{JS} [4]. The divergence between the probability distribution T of the text and the probability distribution A of an automatically generated summary is calculated. The value obtained is a score of the summary information content. Less divergence means greater similarity between $\mathcal{T}$ and $\mathcal{A}$, greater divergence, the opposite.

Despite the good empirical results of the evaluation without references [12, 13, 14], a recurrent criticism towards this kind of algorithms is that nothing prevents using the divergence function between probability distributions in the process of summaries generation.

For example, at each step the algorithm could compute the weight Ω of the sentence f_i, calculated as a measure dependent on the normalized divergence between the probability distributions of the source text $\mathcal{T}$ without the sentence f_i and the involved sentence f_i; f_i would be guaranteed the lowest possible divergence from $\mathcal{T}$ (or the maximum score $\Omega(f_i)$) under any condition. Proceeding in steps, other sentences could be added following a constructive heuristic. This heuristic may generate a summary $\mathcal{A}$ with the k best sentences diverging as little as possible from the source $\mathcal{T}$, as has been suggested by [34]. Other heuristics are also possible.

Supposing that a summarization algorithm G using a sentence weighting Ω with a DPD D. Suppose the divergence D is also used in an evaluation algorithm E. After the generation process, if the summary $\mathcal{A}$ generated by G is

evaluated with the algorithm E, the question is obvious: Can it be guaranteed that the summary $\mathcal{A}$ has the least possible divergence from $\mathcal{T}$? This is a delicate point on which our attention is focused on, because these kinds of heuristics can be blamed of being the "judge" and "part" in the tasks of evaluation and generation of summaries. In the next section we will show that this is not the case.

3 Textual Divergences using Probability Distributions

3.1 Preliminaries

Let the random variable $\mathcal{X}$ be the alphabet (or vocabulary) of a textual document $\mathcal{T}$. A probability distribution of a text $\mathcal{T}$ is a function $T : w \rightarrow p(w), \forall\, w \in \mathcal{X}$. T assigns to each word w in the vocabulary the probability $p(w)$ that this word occurs in $\mathcal{T}$.

This probability $p(w)$ can be estimated by:

$$p(w) = \frac{C_w^{\mathcal{T}}}{|\mathcal{T}|},$$

where $C_w^{\mathcal{T}}$ is the number of occurrences of the word w in the text $\mathcal{T}$ and $|\mathcal{T}| = \sum_w C_w^{\mathcal{T}}$ is the size of the whole document, i.e the total number of occurrences that appear in the text $\mathcal{T}$.

Evaluating a summary using a measure of divergence D is equivalent to calculating the divergence between the probability distribution of the source text $\mathcal{T}$, denoted by T, and the probability distribution of the summary $\mathcal{A}$, denoted by A. We write this as:

$$D(T||A) \quad \text{or} \quad D(T \,||\, \{f_1 \cup \cdots \cup f_k\}), \qquad (1)$$

when the summary consists of k sentences.

In general $D(T||A) > 0$, intuitively, a value close to zero obtained by the equation 1 implies a high similarity of the summary $\mathcal{A}$ in relation to the text source $\mathcal{T}$. Values far from zero imply less similarity of the summary in question. Given a set of summaries $\{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{\mathrm{r}}\}$ with $\mathrm{r} > 2$ is ranked using equation 1, i.e. by measuring their quality (divergence) in relation to the source document.

Generating a summary requires the use of a heuristic or m-steps algorithm. Heuristics can be constructive (by adding selective sentences) or exhaustive (using a combination of all sentences). The use of an exhaustive heuristic is dealt with section 4 and in B.

If the weighting Ω employs a divergence function D applied on the f_i sentences, then generating the summary $\mathcal{A}$ requires an algorithm of a constructive nature.

This involves calculating the weight of subsets of the text (one sentence or a group of sentences) and selecting the best weighted subsets (with minimum divergence) and concatenating them to produce the required summary.

Supposing that in the m constructive step, k sentences have been retained to form part of the summary $\mathcal{A}_m$ of $\mathcal{T}$, then $\mathcal{A}_m = \{f_{m_1} \cup \cdots \cup f_{m_k}\}$. To show that the evaluation of the summary $\mathcal{A}_m$ is not equivalent to the constructive generation of it, it is enough to prove:

$$D(T \,||\, \{f_{m_1} \cup \cdots \cup f_{m_k}\}) \neq D(T||f_{m_1}) + \cdots + D(T||f_{m_k}). \qquad (2)$$

In this article we prove for the asymmetric divergence of Kullback-Leibler [2] and for the symmetrized divergence of Jensen-Shannon [4] that evaluating a summary is different from generating it, even if the same divergence function was used for both tasks.

To prove equation 2 for these DDPs, we need the next three assumptions:

1. The source text consists of at least three sentences, $\mathcal{T} = \{f_1, f_2, \ldots, f_P\}$, with $P \geq 3$.
2. The summary $\mathcal{A}$ consists of at least two sentences and
3. $|\mathcal{A}| < |\mathcal{T}|$.

3.2 DDP Kullback-Leibler

Definition 1 *The divergence, relative entropy or Kullback-Leibler distance between two probability distributions p and q over the same alphabet $\mathcal{X}$ is given by:*

$$D_{KL}(p\|q) = \sum_{w\in\mathcal{X}} p(w)\log_2\frac{p(w)}{q(w)}.$$

Hereafter, for purposes of facilitating reading, we shall agree to write: $\log_2 = \log$, $p(w) = p_w$ and $q(w) = q_w$.

In addition, we also need to establish some conventions: for some words w in $\mathcal{T}$, $C_w^{\mathcal{A}}$ could be zero, because not all words in the text are present in the summary, then $q_w = 0$ and $p_w \log\frac{p_w}{0}$ would be a very large number. For practical reasons, in order to avoid this situation, we can use a smoothing (kind of Laplace, Good-Turing, Back-Off, etc.) that replaces the null values for a small and positive real number [19, 18]. For the case of this demonstration, it suffices to suppose a smoothing where:

$$q_w = \begin{cases} \frac{C_w^{\mathcal{A}}}{|\mathcal{A}|}, \text{ if } w \in \mathcal{A}, \\ \alpha \text{ elsewhere; } 0 < \alpha < 1. \end{cases} \tag{3}$$

The Kullback-Leibler distance is not a metric, because even though it is true that $D_{KL}(p\|q) \geq 0$ with equality if and only if $p \equiv q$, it also occurs that $D_{KL}(p\|q) \neq D_{KL}(q\|p)$ and it does not satisfy the triangle inequality.

Despite this, in the area of automatic summaries, the Kullback-Leibler distance is used to measure the divergence between the probability distribution T of a source document and the probability distribution A of its summary, and it is defined as follows:

$$D_{KL}(T\|A) = \sum_{w\in\mathcal{T}} p_w \log\frac{p_w}{q_w}, \tag{4}$$

where $p_w = \frac{C_w^{\mathcal{T}}}{|\mathcal{T}|}$, q_w is given by the equation 3, $C_w^{\mathcal{T}}$ is the number of occurrences of the word w in $\mathcal{T}$, $C_w^{\mathcal{A}}$ is the number of occurrences of the word w in $\mathcal{A}$, $|\mathcal{T}| = \sum_w C_w^{\mathcal{T}}$ and $|\mathcal{A}| = \sum_w C_w^{\mathcal{A}}$.

Theorem 1 *Let $\mathcal{T}$ be a text with $P \geq 3$ sentences and a probability distribution T. Let f_i and f_j be two sentences in $\mathcal{T}$, and $\mathcal{A} = \{f_i \cup f_j\}$ the summary or concatenation of f_i and f_j with a probability distribution A; then:*

$$D_{KL}(T||\{f_i \cup f_j\}) = D_{KL}(T||f_i) + D_{KL}(T||f_j),$$

occurs only when the sentences f_i and f_j have length zero.

Proof 1 *We will calculate first $D_{KL}(T||f_i \cup f_j)$. To simplify the notation we will write $\mathrm{KL}_w = p_w \log\frac{p_w}{q_w}$. We know that we can decompose the source text as:*

$$\mathcal{T} = (\mathcal{T}\backslash\{f_i \cup f_j\}) \cup (f_i\backslash\{f_i \cap f_j\}) \cup (f_j\backslash\{f_i \cap f_j\}) \cup \{f_i \cap f_j\},$$

then:

$$\begin{aligned} &D_{KL}(T||f_i \cup f_j) \\ &= \sum_{w\in\mathcal{T}\backslash\{f_i\cup f_j\}} \mathrm{KL}_w + \sum_{w\in f_i\backslash\{f_i\cap f_j\}} \mathrm{KL}_w \\ &\quad + \sum_{w\in f_j\backslash\{f_i\cap f_j\}} \mathrm{KL}_w + \sum_{w\in\{f_i\cap f_j\}} \mathrm{KL}_w. \end{aligned} \tag{5}$$

On the other hand:

$$\begin{aligned} &D_{KL}(T||f_i) + D_{KL}(T||f_j) \\ &= \left(\sum_{w\in\mathcal{T}\backslash f_i} KL_w + \sum_{w\in f_i\backslash\{f_i\cap f_j\}} KL_w \right. \\ &\quad \left. + \sum_{w\in\{f_i\cap f_j\}} KL_w \right) \\ &\quad + \left(\sum_{w\in\mathcal{T}\backslash f_j} KL_w + \sum_{w\in f_j\backslash\{f_i\cap f_j\}} KL_w \right. \\ &\quad \left. + \sum_{w\in\{f_i\cap f_j\}} KL_w \right) \\ &= \sum_{w\in\mathcal{T}\backslash f_i} KL_w + \sum_{w\in\mathcal{T}\backslash f_j} KL_w \\ &\quad + \sum_{w\in f_i\backslash\{f_i\cap f_j\}} KL_w + \sum_{w\in f_j\backslash\{f_i\cap f_j\}} KL_w \\ &\quad + \sum_{w\in\{f_i\cap f_j\}} 2KL_w. \end{aligned} \tag{6}$$

Matching the equations 5 and 6, and performing the operations we obtain that:

$$D_{KL}(T||\ \{f_i \cup f_j\}) = D_{KL}(T||f_i) + D_{KL}(T||f_j),$$

implies:

$$\sum_{w\in\mathcal{T}\setminus\{f_i\cup f_j\}} \mathrm{KL}_w = \sum_{w\in\mathcal{T}\setminus f_i} KL_w + \sum_{w\in\mathcal{T}\setminus f_j} KL_w + \sum_{w\in\{f_i\cap f_j\}} KL_w. \quad (7)$$

As we can see, the sum of the left side of the equation 7 is calculated on the set $\mathcal{T}\setminus\{f_i\cup f_j\}$ while the sum of the right side is calculated on the whole set $\mathcal{T}$. Giving that $(f_1 \cup f_2) \subset \mathcal{T}$, the only way it occurs that:

$$\mathcal{T}\setminus\{f_i \cup f_j\} = \mathcal{T},$$

is when:

$$f_i \cup f_j = \emptyset.$$

From the above it is concluded that:

$$D_{KL}(T||\{f_i \cup f_j\}) = D_{KL}(T||f_i) + D_{KL}(T||f_j) \Leftrightarrow |f_i| = |f_j| = 0.$$

this is when the sentences f_i and f_j have length zero.

QED

3.3 DDP using Jensen-Shannon

The Jensen-Shannon divergence D_{JS} [4], or symmetrized Kullback-Leibler distance between two probability distributions p on the alphabet $\mathcal{X}$ and q on the alphabet $\mathcal{Y}$, where $\mathcal{Y} \subseteq \mathcal{X}$ is given by:

$$D_{JS}(p\|q) = \frac{1}{2}\left\{\sum_{x\in\mathcal{X}} p(x)\log\frac{2p(x)}{p(x)+q(x)} + \sum_{x\in\mathcal{Y}} q(x)\log\frac{2q(x)}{p(x)+q(x)}\right\}.$$

With the same convention (3) used in the divergence D_{KL}.

Use D_{JS} to measure the divergence between the probability distribution T of a text $\mathcal{T}$ and the probability distribution A of a summary $\mathcal{A} = \{f_i \cup f_j\}$ implies:

$$D_{JS}(T\|A) = \frac{1}{2}\left\{\sum_{w\in\mathcal{T}} p_w \log\frac{2p_w}{p_w+q_w} + \sum_{w\in\mathcal{A}} q_w \log\frac{2q_w}{p_w+q_w}\right\},$$

with the same meanings for T, A, $|\mathcal{T}|$, $|\mathcal{A}|$, p_w, $C_w^{\mathcal{T}}$ and $C_w^{\mathcal{A}}$ (defined for the equation (4)) and in the Theorem 1.

Theorem 2 *Let $\mathcal{T}$ be a text with $P \geq 3$ sentences and a probability distribution T. Let f_i and f_j be two sentences in $\mathcal{T}$, and $\mathcal{A} = \{f_i \cup f_j\}$ the summary or concatenation of f_i and f_j with a probability distribution A. Then:*

$$D_{JS}(T||\{f_i \cup f_j\}) = D_{JS}(T||f_i) + D_{JS}(T||f_j),$$

occurs only when the sentences f_i and f_j have length zero.

Proof 2 *From now on, we will write:*

$$P_w = p_w \log\frac{2p_w}{p_w+q_w}; \quad Q_w = q_w \log\frac{2q_w}{p_w+q_w}.$$

We calculate first :

$$\begin{aligned} D_{JS}(T||f_i\cup f_j) &= \frac{1}{2}\left\{\sum_{w\in\mathcal{T}} P_w + \sum_{w\in\mathcal{A}} Q_w\right\} \\ &= \frac{1}{2}\left\{\sum_{w\in\mathcal{T}\setminus\{f_i\cup f_j\}} P_w + \sum_{w\in\mathcal{A}\setminus\{f_i\cup f_j\}} Q_w\right\} \\ &+\frac{1}{2}\left\{\sum_{w\in f_i\setminus\{f_i\cap f_j\}} P_w + \sum_{w\in f_i\setminus\{f_i\cap f_j\}} Q_w\right\} \quad (8) \\ &+\frac{1}{2}\left\{\sum_{w\in f_j\setminus\{f_i\cap f_j\}} P_w + \sum_{w\in f_j\setminus\{f_i\cap f_j\}} Q_w\right\} \\ &+\frac{1}{2}\left\{\sum_{w\in\{f_i\cap f_j\}} P_w + \sum_{w\in\{f_i\cap f_j\}} Q_w\right\}. \end{aligned}$$

On the other hand, we have:

$$
\begin{aligned}
&D_{JS}(T||f_i) + D_{JS}(T||f_j) \\
&= \frac{1}{2}\left\{ \sum_{w\in\mathcal{T}\setminus f_i} P_w + \sum_{w\in\mathcal{A}\setminus f_i} Q_w \right. \\
&\quad + \sum_{w\in f_i\setminus\{f_i\cap f_j\}} P_w + \sum_{w\in f_i\setminus\{f_i\cap f_j\}} Q_w \\
&\quad \left. + \sum_{w\in\{f_i\cap f_j\}} P_w + \sum_{w\in\{f_i\cap f_j\}} Q_w \right\} \\
&\quad + \frac{1}{2}\left\{ \sum_{w\in\mathcal{T}\setminus f_j} P_w + \sum_{w\in\mathcal{A}\setminus f_j} Q_w \right. \\
&\quad + \sum_{w\in f_j\setminus\{f_i\cap f_j\}} P_w + \sum_{w\in f_j\setminus\{f_i\cap f_j\}} Q_w \\
&\quad \left. + \sum_{w\in\{f_i\cap f_j\}} P_w + \sum_{w\in\{f_i\cap f_j\}} Q_w \right\}.
\end{aligned} \tag{9}
$$

Matching equations (8) and (9), regrouping terms and doing the necessary operations we obtain:

$$
\begin{aligned}
&\frac{1}{2}\left\{ \sum_{w\in\mathcal{T}\setminus\{f_i\cup f_j\}} P_w + \sum_{w\in\mathcal{A}\setminus\{f_i\cup f_j\}} Q_w \right\} \\
&= \frac{1}{2}\left\{ \sum_{w\in\mathcal{T}\setminus f_i} P_w + \sum_{w\in\mathcal{A}\setminus f_i} Q_w \right. \\
&\quad + \sum_{w\in\mathcal{T}\setminus f_j} P_w + \sum_{w\in\mathcal{A}\setminus f_j} Q_w \\
&\quad \left. + \sum_{w\in\{f_i\cap f_j\}} P_w + \sum_{w\in\{f_i\cap f_j\}} Q_w \right\}.
\end{aligned} \tag{10}
$$

As in Theorem 1, we can see that the sum of the left side of equation 10 is calculated on the set $\mathcal{T}\setminus\{f_i \cup f_j\}$ while the sum of the right side is calculated on the whole set $\mathcal{T}$. Giving that $(f_1 \cup f_2) \subset \mathcal{T}$, the only way it occurs that:

$$\mathcal{T}\setminus\{f_i \cup f_j\} = \mathcal{T},$$

is when: $f_i \cup f_j = \emptyset$

From the above it is concluded:

$$
\begin{aligned}
D_{JS}(T \,||\, \{f_i \cup f_j\,) &= D_{JS}(T||f_i) + D_{JS}(T||f_j) \\
&\Leftrightarrow |f_i| = |f_j| = 0,
\end{aligned}
$$

this is when the sentences f_i and f_j have length zero.
QED

4 Discussion and Conclusions

The main contribution of this article is to show that a DDP used in a summary evaluator algorithm can not be used in a generator algorithm to construct a summary without introducing a bias. In other words, in spite of using the same DDP for both construction (aggregation of sentences using heuristics) and for the evaluation of a summary, it can not be guaranteed that the evaluation of the given summary can obtain the minimum divergence with respect to the source. Hence, it is not unreasonable to use the same DDP in both tasks.

In this article, we have shown that under the three next conditions:

1. The divergence definition D uses D_{KL} or D_{JS},
2. The original document $\mathcal{T}$ contains at least 3 sentences, and
3. The summary is obtained by extraction and contains at least 2 sentences, $\mathcal{A} \subset \mathcal{T}$ and $|\mathcal{A}| < |\mathcal{T}|$,

it happens that $D(T||\{f_i \cup f_j\}) \neq D(T||f_i) + D(T||f_j)$.

We have tested it with the Kullback-Leibler probability distributions D_{KL} and Jensen-Shannon D_{JS}. In the case of the divergence D_{KL}, given its asymmetry, it can be considered from the distribution of probabilities T of the text compared to the distribution of probabilities A of the summary and vice versa. In this article, we only consider the case $D_{KL}(T||A)$, but a similar analysis can show that the Theorem 1 is also fulfilled in the case $D_{KL}(A||T)$.

However, an additional case must be considered. In the extraction paradigm, a summary generator G^* could use an exhaustive

combinatorial approach combined with an *a posteriori* divergence evaluation E. That is, an algorithm $G^*(\mathcal{T})$ could theoretically and exhaustively produce the set $\{\mathcal{A}\}$ of all the r possible extracts and evaluates each one of them using $E(D(T||A_i)); i = 1, 2, \ldots, \mathrm{r}$, thus retaining the subset $\{a\} \subseteq \mathcal{A}$ summaries that minimize divergence D. If the subset of summaries $\{a\}$ is automatically evaluated using the same DDP D, each of these summaries will have the guarantee of obtaining the best evaluation for divergence in relation to $\mathcal{T}$. Yet, in practice this idea would not work. In fact, the total number r of extracts having S sentences generated from a document $\mathcal{T}$ having P sentences, is given by the binomial coefficient:

$$r = \binom{P}{S} = \frac{P!}{S!(P-S)!}.$$

This number can be large enough even for moderate values of P and S. For example, from a source having P = 40 sentences and a compression rate $\rho = 20\%$, the number of possible extracts of S = 8 sentences is $\mathrm{r} = 76'904,685$. Generating and evaluating them exhaustively in order to keep less divergent summaries is not practical.

Avoiding the exhaustive algorithm, the argument that we have presented eliminates the main criticism towards summarization evaluations without references, in the sense that using DDP during generation and evaluation processes constitutes cheating. We can state that, under conditions mentioned above, evaluating and constructing a summary are two mathematically different tasks, although the DDP used in both is the same.

A Documents Representation as Probability Distribution

Preprocessing

Formally, a text is composed of paragraphs, sentences, words and punctuation marks. In NLP, a classic preprocessing [19, 18] allows the text to be split into sentences and tokens. Preprocessing can eliminate non-informative tokens (articles, conjunctions, some adjectives, auxiliary verbs, punctuation, etc.). A deeper preprocessing can decrease the variability of the morphological and graphical forms of the remaining tokens using a normalization of the vocabulary [21]. The normalization can be carried out using a lemmatization (infinitive verbs, nouns in masculine singular, etc.) [19], a stemming (pseudo-roots of the words) [28] or ultra stemming (truncated words to the first characters). The goal is to transform words into canonical forms where the number of occurrences of standardized tokens allows you to conveniently represent documents in a small space.

Example 1. Let the text be: *"The little prince had a lamb and a rose. Lambs are nice animals. The little prince has no friends except the fox and the lamb... poor prince!"*. Text can be preprocessed to produce the tokenized and normalized output: $\mathcal{T}$ = {[prince have rose lamb], [nice animal lamb], [prince have friend fox lamb prince]}. In this representation, the tokens "prince" and "lamb" occur 3 times, "have" occurs 2 times and the other words occur 1 time. The vocabulary of $\mathcal{T}$ has 8 different tokens: {prince, have, lamb, rose, animal, nice, friend, fox}.

Vector Space Model and Document Representation

This representation (and its variants) [19, 18] is very suitable and extensively used in Extraction and Information Retrieval tasks, as well as in NLP. The representation can be formalized as follows:

Let $\mathcal{T}$ be a document consisting of a set of P sentences. Let f_i be a sentence $\in \mathcal{T}$, consisting of a set of normalized tokens $w_{ij}, j = 1, 2, \ldots, n$. Each token w_{ij} has a number of occurrences associated with the document $\mathcal{T}$, denoted by $C^{\mathcal{T}}_{w_{ij}}$; $C^{\mathcal{T}}_{w_{ij}} \in \mathbb{N}$. The number of occurrences of the token w_{ij} can also be counted in each sentence f_i as $C^{f_i}_{w_{ij}}$ and can be weighted by $C^{T}_{w_{ij}}$ combined with the Inverse Document Frequency ($IDF_{w_{ij}}$) in a text $\mathcal{T}$: $IDF_{w_{ij}} = \log \frac{P}{d_{wij}}$; where P is the number of sentences of $\mathcal{T}$ and $d_{w_{ij}}$ is the number of sentences where w_{ij} appears. In this way, words can be weighted by $C^{\mathcal{T}}_{w_{ij}} \times \log \frac{P}{d_{w_{ij}}}$.

In this paper, tokens are not weighted, but their estimated probability is:

$$p(w) = \frac{C_w^{\mathcal{T}}}{\sum_w C_w^{\mathcal{T}}}.$$

In the vector model the order of the tokens in each sentence is not important and the sentences are treated as a bag-of-words.

Example 2. In the above example, token occurrences of "prince" and "lamb" in the text $\mathcal{T}$ and in the sentences f_1, f_2 y f_3 are: $C^{\mathcal{T}}_{\text{prince}} = 3$; $C^{f_1}_{\text{prince}} = 1$, $C^{f_2}_{\text{prince}} = 0$; $C^{f_3}_{\text{prince}} = 2$; $C^{\mathcal{T}}_{\text{lamb}} = 3$; $C^{f_{1,2,3}}_{\text{lamb}} = 1$.

The probability distribution of the text $\mathcal{T}$, ($|\mathcal{T}|$=13) is: $\mathcal{T} = (\text{prince} = \frac{3}{13}, \text{have} = \frac{2}{13}, \text{lamb} = \frac{3}{13}, \text{rose} = \frac{1}{13}, \text{animal} = \frac{1}{13}, \text{nice} = \frac{1}{13}, \text{friend} = \frac{1}{13}, \text{fox} = \frac{1}{13})$.

The probability distribution of the first sentence f_1 ($|f_1| = 4$) is: $f_1 = (\text{prince} = \frac{1}{4}, \text{have} = \frac{1}{4}, \text{lamb} = \frac{1}{4}, \text{rose} = \frac{1}{4})$.

Recently, we introduced continuous representation spaces of words (word embeddings) [22, 9]. These techniques are also employed in ATS, but in this paper we will discuss only discrete representations.

B Automatically Generation of Summaries

The classical extractive paradigm of generation of summaries [15, 1, 3], is currently dominant in the literature for several reasons: it is easy to program, reproducible, language independent, etc. [16, 34]. The result is not an abstract in the human sense of the term, but an "extract" composed of important sentences from the source. However, recent research focuses on the generation of true abstracts. This task requires specialized modules to generate, fuse and/or compress sentences.

After a preprocessing and representation in a suitable space [4], this paradigm involves the selection and extraction of the k most representative sentences belonging to the document. This set of sentences will be denoted A and is calculated through a weighting Ω (or score normalized between $[0, 1]$) of the individual sentences from the source document $\mathcal{T}$: the sentences having high scores are aggregated to A, the rest are probably eliminated.

There are several types of summary-generating algorithms (*summarization heuristics*):

1. Baselines (Random, Lead/First base, length of the sentences, etc.) [16].
2. Graph Heuristics [5, 20],
3. Statistical/Neural Heuristics, etc. [10, 16, 35, 9], etc.
4. Linguistic Heuristics [26, 16], etc.
5. Mixed Linguistic-Statistical Heuristics [31, 24], etc.
6. Other heuristics. In "Other heuristics" we have two types.
   - — The first one consists in exhaustively generating, from a text $\mathcal{T}$ having P sentences, the set $\{A\}$ of all the summaries having S sentences. Each summary of $\{A\}$ is then evaluated according to its informative content and the most informative subset $\{a\} \subseteq \{A\}$ is retained.
   - — The second one supposes that the existence of algorithms for the summarization of type 2), 3), 4), 5) or their combinations whereby the sentence's weighting function Ω can also be used in an evaluation algorithm. In particular Ω can be a DPD.

C Evaluating Summaries

The DUC/TAC conferences [5] have established an empirical framework for the evaluation of summaries since 2001 [24]. In this section, we will do a quick review of the three types of intrinsic evaluation of text summaries: manual, automatic using references and automatic without references.

[4] Space where words can be weighted or not.

[5] See websites: `http://duc.nist.gov` and `http://www.nist.gov/tac`.

Manual and Automatic Evaluations using References

Manual evaluation involves direct reading and analysis by human judges [16]. Of course, there is a high degree of subjectivity in the task of evaluation defined in this way. At the granularity level of the sentence, there is the problem of whether two different sentences have the same meaning. Anyway, the objective is dealing with the semantics of the summary, at this level, the concordance between judges is not guaranteed at all. However, a strong hypothesis is that humans know how to generate and understand abstracts, their judgments are considered a reference for evaluating artificial summaries. In particular, the assessment of linguistic quality is done only manually [8]. Several methods using human evaluators have been proposed [23]:

1. Readability: Qualitative score of linguistic quality.
2. Responsiveness: Qualitative score of overall responsiveness to the given task.
3. Pyramid: A quantitative measure of content [25, 27].

In NLP, n-grams of words are a convenable way of represent texts. A n-gram is a sequence of n words, $n = 1, 2, ...$ whereby the words may or may not be consecutives. In particular, if $n = 2$, we speak of bigrams or sequences of two words: $w_i w_j$. n-grams representation is a suitable way of capture lexical fragments beyond the word. Automatic evaluations that use references to calculate the intersection of n-grams of words between a candidate summary and a set of human-based reference abstracts. This intersection normalized between $[0, 1]$ on the set of n-grams is a measure of recall.

An automated evaluation system currently plebiscited by the scientific community is ROUGE (Recall-Oriented Understanding Gist Evaluation) [11]. Various systems (ROUGE-1,2, SU4, Basic Elements BEwT-E [6], CLASSY1, DemokritosGR, uOttawa3, CS-IITH1-4, etc.) presented in the TAC 2011 AESOP task (Automatically Evaluating Summaries of Peers) show a high correlation with manual evaluations [30]. In these campaigns, ROUGE has shown to be the best automatic method using human references [29].

Automatic Evaluation without Human References

This type of evaluation requires using a divergence between two probability distributions. The Kullback-Leibler divergence D_{KL} [2] and the Jensen-Shannon divergence D_{JS} [4] have been extensively used in this task. The DPD between the text T and a summary A represents a measure of the summary information content: less divergence implies a greater similarity between T and A. An immediate advantage is that these measures do not require human references to be calculated. This point avoids the subjectivity present in human references. The evaluation algorithms (SIMetrix [6], Fresa[7], etc.) that use DPD have given very good results [7, 12, 13, 14], comparable with results produced using manual or automatic methods with references. In addition, these algorithms have been tested in several automatic text summarization multi-document/multi-lingual tasks.

References

1. **Baxendale, P. (1958).** Machine-made index for technical literature: an experiment. *IBM Journal of Research Development*, Vol. 2, No. 4, pp. 354–361.
2. **Cover, T. & Thomas, J. (1991).** *Elements of information theory*. Wiley.
3. **Edmundson, H. (1969).** New methods in automatic extraction. *Journal of the Association for Computing Machinery*, Vol. 16, No. 2, pp. 264–285.
4. **Endres, D. & Schindelin, J. (2003).** A new metric for probability distributions. *IEEE Trans. Inform. Theory*, Vol. 49, No. 7, pp. 1858–1860.
5. **Erkan, G. & Radev, D. (2004).** LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, Vol. 22, pp. 457–479.

[6] http://homepages.inf.ed.ac.uk/alouis/IEval2.html
[7] http://fresa.talne.eu/

6. **Hovy, E., Lin, C.-Y., Zhou, L., & Fukumoto, J. (2006).** Automated Summarization Evaluation with Basic Elements. *5th International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy.
7. **Jianhua, L. (1991).** Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, Vol. 37, pp. 145–151.
8. **Jing, H., Barzilay, R., McKeown, K., & Elhadad, M. (1998).** Summarization evaluation methods: Experiments and analysis. *AAAI Symposium On Intelligent Summarization*, pp. 60–68.
9. **Kågebäck, M., Mogren, O., Tahmasebi, N., & Dubhashi, D. (2014).** Extractive summarization using continuous vector space models. *2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC) EACL, April 26-30, 2014 Gothenburg, Sweden*, pp. 31–39.
10. **Kupiec, J., Pedersen, J., & Chen, F. (1995).** A trainable document summarizer. *18th Conference ACM Special Interest Group on Information Retrieval (SIGIR'95)*, ACM Press, New York, Seattle, WA, USA, pp. 68–73.
11. **Lin, C.-Y. (2004).** ROUGE: A Package for Automatic Evaluation of Summaries. **Marie-Francine Moens, S. S.**, editor, *Text Summarization Branches Out: ACL-04 Workshop*, Barcelona, Spain, pp. 74–81.
12. **Louis, A. & Nenkova, A. (2008).** Automatic Summary Evaluation without Human Models. *TAC'08 (NIST)*.
13. **Louis, A. & Nenkova, A. (2009).** Automatically Evaluating Content Selection in Summarization without Human Models. *Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Singapore, pp. 306–314.
14. **Louis, A. & Nenkova, A. (2013).** Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, Vol. 39, No. 2, pp. 267–300.
15. **Luhn, H. (1958).** The automatic creation of literature abstracts. *IBM Journal of Research and Development*, Vol. 2, No. 2, pp. 159–165.
16. **Mani, I. (2001).** *Automatic summarization*. John Benjamins Publishing Co.
17. **Mani, I. & Mayburi, M. (1999).** *Advances in automatic text summarization*. MIT Press, Cambridge.
18. **Manning, C., Raghavan, P., & Schütze, H. (2008).** *Introduction to Information Retrieval*. Cambridge University Press.
19. **Manning, C. & Schütze, H. (1999).** *Foundations of statistical natural language processing*. The MIT Press, Cambridge, Massachusetts.
20. **Mihalcea, R. (2004).** Graph-based ranking algorithms for sentence extraction, applied to text summarization. *ACL'04 on Interactive poster and demonstration sessions*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 181–184.
21. **Mikheev, A. (1999).** Periods, capitalized words, etc. *Computational Linguistics*, Vol. 28, No. 3, pp. 289–318.
22. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013).** Distributed representations of words and phrases and their compositionality. pp. 3111–3119.
23. **NAACL-HLT (2012).** Workshop on evaluation metrics and system comparison for automatic summarization. *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, volume 1, Montreal, Quebec, Canada, pp. 52.
24. **Nenkova, A. & McKeown, K. (2011).** Automatic summarization. *Foundations and Trends in Information Retrieval*, Vol. 5, No. 2-3, pp. 103–233.
25. **Nenkova, A., Passonneau, R., & McKeown, K. (2007).** The pyramid method: Incorporating human content selection variation in summarization evaluation. *ACM Transactions on Speech and Language Processing*, Vol. 4, No. 2, pp. 23.
26. **Ono, K., Sumita, K., & Miike, S. (1994).** Abstract generation based on Rhetorical Structure extraction. *15th ICCL*, Kyoto, pp. 344–348.
27. **Passonneau, R., Nenkova, A., McKeown, K., & Sigleman, S. (2005).** Applying the Pyramid Method in DUC 2005. *DUC'05 (HLT/EMNLP)*.
28. **Porter, M. (1980).** An algorithm for suffix stripping. *Program*, Vol. 14, No. 3, pp. 130–137.
29. **Rankel, P. A., Conroy, J. M., Dang, H. T., & Nenkova, A. (2013).** A Decade of Automatic Content Evaluation of News Summaries: Reassessing the State of the Art. *ACL (2)*, pp. 131–136.
30. **Rankel, P. A., Conroy, J. M., Dang, H. T., & Nenkova, A. (2013).** A decade of automatic content evaluation of news summaries: Reassessing the state of the art. *51st Annual Meeting of*

the Association for Computational Linguistics, ACL'2013, Sofia, Bulgaria, Volume 2: Short Papers, pp. 131–136.

31. **SanJuan, E., Ibekwe-SanJuan, F., Torres-Moreno, J.-M., & Velázquez-Morales, P. (2007).** Combining Vector Space Model and Multi Word Term Extraction for Semantic Query Expansion. *12th NLDB*, volume 4592 of *LNCS*, Springer-Verlag, Paris, France, pp. 252–263.
32. **Spärck-Jones, K. (1997).** Summarizing: Where are we now? Where should we go. **Mani, I. & Maybury, M.**, editors, *ACL/EACL'97 Workshop on Intelligent Scalable Text Summarization*, Madrid, Spain, pp. 1.
33. **Spärck-Jones, K. (1999).** *Automatic summarizing: factors and directions*. The MIT Press.
34. **Torres-Moreno, J. (2014).** *Automatic text summarization*, volume 1. John Wiley & Sons.
35. **Torres-Moreno, J.-M., Velázquez-Morales, P., & Meunier, J.-G. (2001).** Cortex: un algorithme pour la condensation automatique des textes. *ARCo'01*, volume 2, Lyon, France, pp. 365–366.

Article received on 22/06/2020; accepted on 23/07/2020. Corresponding author is Juan Manuel Torres Moreno.