Automated Snippet Generation for Online Advertising

Stamatina Thomaidou[1], Ismini Lourentzou[1,2], Panagiotis Katsivelis-Perakis[1,3], and Michalis Vazirgiannis[1,4]
[1]Athens University of Economics and Business, [2]University of Illinois at Urbana-Champaign, [3]Harokopion University of Athens, [4]Ecole Polytechnique, Paris
{thomaidous, katsivelhsp}@aueb.gr, lourent2@illinois.edu, mvazirg@lix.polytechnique.fr

ABSTRACT

Products, services or brands can be advertised alongside the search results in major search engines, while recently smaller displays on devices like tablets and smartphones have imposed the need for smaller ad texts. In this paper, we propose a method that produces in an automated manner compact text ads (promotional text snippets), given as input a product description webpage (landing page). The challenge is to produce a small comprehensive ad while maintaining at the same time relevance, clarity, and attractiveness. Our method includes the following phases. Initially, it extracts relevant and important n-grams (keywords) given the landing page. The keywords reserved must have a positive meaning in order to have a call-to-action style, thus we attempt sentiment analysis on them. Next, we build an Advertising Language Model to evaluate phrases in terms of their marketing appeal. We experiment with two variations of our method and we show that they outperform all the baseline approaches.

Categories and Subject Descriptors

I.7 [**Computing Methodologies**]: Document and Text Processing

General Terms

Algorithms, Experimentation, Performance

Keywords

textual advertising, online advertising, sponsored search, automated ad-text generation

1. INTRODUCTION AND MOTIVATION

Sponsored search has proposed a new way of targeted and textual advertising, which has evolved into a billion dollar industry the past few years. The general idea behind sponsored search is that products, services or brands can be advertised alongside the search results in major search engines, matching the input queries of the users. When a person types a query, it is matched to particular keywords which trigger a particular set of ads. For each ad we recognize four integral parts: a. *The title of the ad*, where one can find the name of the advertised product, service or brand b. *The ad-text snippet*, which is a short advertising text (usual limit 70 characters) c. *The display url*, which most of the times is the url of the main website and d. *The landing page*, which is the exact url of a product, service or brand description page inside the main website.

In this paper, we propose an engine which produces compact ad-text snippets in an automated and massive manner given a product landing page as input. Such a system aims at facilitating the process of online advertising. The main notion is to provide an efficient solution for online marketing campaigns that feature large websites or shops that can be considered as large online product catalogues. These sites may include hundreds or thousands products or services that each one of them need to be promoted through a text ad. At the same time, there is an emerging need for promotion through channels that require more and more short promotional text like interfaces on tablets and smartphones. In this way, our method contributes with the automated generation of compact but comprehensive ad text.

Our proposed system functions on the following phases:

1. *Information Extraction* for mining the most important product or service keywords.
2. *Sentiment Analysis* for keeping the most positive phrases that will have a good impact on the product image.
3. *Natural Language Generation* for constructing a good form of the final ad-text sentences.

2. RELATED WORK

The literature is rich on proposed information retrieval methods in the area of sponsored search and textual advertising focusing mainly on keyword selection. Ravi et al.[6] present a method for bid phrases through a monolingual translation model, the use of a probabilistic framework, and a language model. In the same context, Broder et al.[2] try to predict new phrases for an ad from a real life ad corpus. Regarding the automated ad-text generation process, to the best of our knowledge, this issue remains still an open problem as mentioned in [4]. Bartz et al.[1] approach the issue by using strict ad templates with slots reserved for bidding terms that need to be carefully reformatted to suit the given template. Fujita et al.[3] attempt to generate shop-specific listing ads (in japanese) by reusing previously written text ads for them.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.
Copyright 2013 ACM 978-1-4503-2263-8/13/10 ...$15.00.
http://dx.doi.org/10.1145/2505515.2507876 .

3. PROPOSED METHOD

3.1 Information Extraction

In this phase the goal is to gather information about the promoted product. Our only source for this kind of information is the landing page of the product, from which we need to obtain the most important parts that must be conveyed into the generated corresponding ad-text. We therefore employ an unsupervised summarization method based on the work of Ganesan et al.[5]. The method relies on the construction of a unigram model, while extending it with higher order n-grams until there is no further production of new n-grams available. At every step of forming newer n-grams, non-promising candidates are pruned by calculating a scoring metric named as *representativeness*, introduced in [5], that captures how well a phrase represents the given webpage.

Representativeness score is essentially a modified point-wise mutual information metric. Words used in keyphrases should not only be frequent in text, but also strongly associated. Point-wise mutual information measures strength of association between words. Bigram point-wise mutual information is defined as follows:

$$pmi(w_i, w_j) = \log_2 \frac{p(w_i, w_j)}{p(w_i)p(w_j)} \tag{1}$$

where $p(w)$ is the probability of a unigram w occurence inside the webpage. Using formula (1), we redefine the numerator of the calculation by multiplying in advance the probability of the co-occurence of the two unigrams with the frequency of the bigram that these unigrams form. This "weight assignment" in the numerator rewards well associated words with high co-occurence. The modified point-wise mutual information formula is defined as follows:

$$pmi'(w_i, w_j) = \log_2 \frac{p(w_i, w_j)c(w_i, w_j)}{p(w_i)p(w_j)} \tag{2}$$

where $c(w_i, w_j)$ is the frequency of the bigram that w_i and w_j form.

Then, the *representativeness* score of each n-gram is the sum of the modified point-wise mutual information of every bigram that this n-gram contains, normalized by the total unigram length. The formula of the representativeness score is presented as:

$$Representativeness(w_1, ..., w_n) = \frac{1}{n} \sum_{i=0, j<i}^{n} pmi'(w_i, w_j) \tag{3}$$

Information extraction phase consists of two main parts: extracting the product name and extracting the keyphrases. Considering a specific webpage, we begin our process by extracting a set of unigrams U, while removing unigrams with frequency equal to 1. Moreover, the median frequency m of the remaining unigrams is calculated. The next step is the formation of a list that contains bigrams and their representativeness score.

We exploit the information in the html title tag in order to find the product name. By creating a unigram list for the title, we gradually form higher order n-grams with the use of two thresholds. At first, we find the first unigram in the title that has higher frequency than m. We proceed on forming a higher order n-gram, by concatenating unigrams that have each time frequency higher than m. When the formulation of this n-gram stops, which means that we have reached a unigram with lower frequency, we use the seed n-gram as input for another iterative process: gradually concatenating each unigram in the title that succeeds the seed n-gram until the formulated by this procedure n-gram has representativeness score lower than the threshold t (after parameter tuning, $t = 1.8$). The result of this method is the product name. In order to avoid duplicate appearances of the product name in our ads, unigrams that can be found in it are removed from our set U, as well as from our bigram set B.

We proceed by extracting the keyphrases. Using n-grams from B as seed candidates, we concatenate each candidate that has representativeness score higher than zero with another n-gram that shares an overlapping word, meaning that the ending word in $ngram_1$ should overlap with the starting word in $ngram_2$. In addition, $ngram_1$ should not be a "mirror" of $ngram_2$. Furthermore, we use cosine similarity to avoid redundancies in our candidates list. Specifically, before adding a candidate phrase X to the list, we check whether there is another phrase in the list that is similar to X. Candidates should have a cosine similarity lower than a threshold y (after parameter tuning, $y=0.7$). If a similar phrase in our candidates list is found then we keep the candidate that has a higher representativeness score. We repeat this iterative process until no possible expansion of our candidates list.

3.2 Sentiment Analysis

Sentiment analysis in our method aims to determine the contextual polarity of a phrase. The vast majority of the current websites that refer to a large available set of products or services (e.g. Amazon, eBay, etc.) offer the possibility for a client to write a *review* or a *comment* for any purchased product. Therefore opinion mining may play an important role in our approach by separating the negative phrases from the positive ones, providing a set of phrases that could potentially be used for marketing purposes.

In this task, the Amazon Sentiment Dataset[1] was used. The dataset consists of Amazon reviews with a 5-star rating. Since this dataset does not contain any neutral reviews (3 stars), we could easily form two categories: positive reviews and negative reviews, tagged **'pos'** or **'neg'** respectively. Separating this dataset into train set and test set, we train a simple Naive Bayes classifier in order to have a first look at the support of the sentiment analysis procedure. For this classification task we experiment in feature selection. Initially, all words in each review were our features. In the next experiments we select the top k informative words as features, by measuring the Information Gain of each word. Using all words as features provided better accuracy in the test set than choosing the top k informative words, thus we proceeded our classification model with all words. Finally, we removed all extracted keyphrases that were classified by our model as negative.

3.3 Natural Language Generation

3.3.1 Content Determination

Our method starts with the list of product features which is the output of the Information Extraction phase (IE). From the n-grams obtained, we need to choose only a few best to be included in the ad-text, due to length and significance

[1]Large-Scale Datasets for Sentiment Analysis `http://mst.cs.drexel.edu/datasets/SentimentDatasets/SentimentDatasets`

restrictions. To do so, we calculate the mean score of the entire set and then choose the n-grams with score higher or equal to the mean. Those with length greater than 70 characters are also removed from the final set. This limit is not arbitrary. Search engines e.g. Google allow text ads with this character limitation.

The method continues with trimming the n-gram set by performing two grammatical checks. At first, n-grams that contain verbs (in any form or tense) are removed from the n-gram list, as they do not fit our chosen realization template. The second check eliminates n-grams of low readability; n-grams that contain sequences of five or more nouns in a row and might have been erroneously extracted from the landing page. To achieve these, we made use of the Stanford Part-of-speech Tagger[2].

3.3.2 Sentence Planning and Realization

During this phase, we build *permutations* of the best n-grams in order to add them to the final representation, assuring the character limitation mentioned before.

At this point, it is essential to picture the final representation of the snippet as a collection of empty slots. The information that needs to be conveyed includes: the product name, a feature sequence about the product and its price (optionally). Our method proposes two simple sentence templates which provide the necessary slots:

$$\langle\text{product name}\rangle \text{ with } \langle\text{feature set}\rangle \ \langle\text{price}\rangle \tag{a}$$

$$\langle\text{feature set}\rangle \ \langle\text{price}\rangle \tag{b}$$

To generate possible candidate ad snippets, we first fill in the <feature set> slot with all feature sequences generated above. We add the extracted product name from the IE phase to the corresponding slot of (a), once again with respect to the size of the generated sentence. In addition, we concat the product price - if available, provided that there is space left for this addition.

3.3.3 Candidate Pruning

The previous stage often results in a large number of candidates, as a consequence of the number of feature sets or the length of the other features. We consider the problem of pruning the output candidate set, in pursuance of retrieving a small amount of high-quality candidates. The idea here is to rank candidates using two proposed ranking functions: the Information Ranking Function and the Readability Ranking Function.

Information Ranking Function It is not uncommon that some generated candidates in the result set do not include the most significant product features, i.e. highest-score n-grams which were extracted from the landing page. Moreover, some others do not utilize the 70-character space efficiently, resulting in extremely small and inadequate ads. Therefore we implement a ranking function that measures the overall Information Gain from the snippet and penalizes those with little utilization of space. For the second template (b), given a candidate c with s_i being the scores of the product features included in it, n the number of features and $l(c)$ the length of the candidate, the value of the function I_i is:

$$I_i(c) = \frac{1}{n + \frac{70}{l(c)}} \sum_{i=0}^{n} s_i \tag{4}$$

For the first template (a), the appearance of the product name is considered to be very significant for the general completeness of an advertising text. The idea here is to promote candidates that feature the product name by taking it into account as a product feature with score equal to the maximum of the set. To meet this standard, we adapt the Information Ranking Function as follows:

$$I_i(c) = \frac{1}{n + \frac{70}{l(c)}} \max_{0<i<n} s_i + \sum_{i=0}^{n} s_i \tag{5}$$

Readability Ranking Function The Readability Ranking Function awards candidates that fulfill two important conditions: they are readable according to advertising standards and grammatically correct. It aims at eliminating any errors or abnormalities in the content of candidates that surface during the Content Determination stage of our NLG engine. It also evaluates the final result of the generation process by answering to whether each candidate can be viewed as an ad.

This measure fully relies on the application of an Advertising Language Model. For this purpose, we had to build an ad dataset of 47.984 unique ads obtained from major search engines. The tested search queries were all the nodes of the Google Products Taxonomy[3], which is a taxonomy of 21 major product categories and their sub-categories. We expanded these phrases by selecting similar categories and keywords from the Google Keyword Suggestion Tool and used them both as broad as well as phrase match queries (i.e. using quotes or not). Each query returned a number of ads, from which we kept all integral parts, along with the path of the query that triggered them in the taxonomy. We feed SRILM Toolkit[4] with the snippets as input corpus. We built a language model based on trigrams and on the Kneser-Ney discounting method, which was used by our engine to assign scores on the generated candidates. Concretely, for each candidate we keep its logarithmic probability as the value of the Readability Ranking Function, which is an indication of the likelihood that a given candidate will occur, according to the language model.

It is important to notice that each ranking function produces scores of different magnitude. To combine these scores we used the Min-Max Normalization Technique to bring all scores in the [0,1] range and find those candidates with a better mean than others. Depending on the number of candidates that we want to keep, we can prune after a certain threshold.

4. EXPERIMENTS AND EVALUATION

In order to examine the potential of the proposed method, we experimented with variations of our principal method. In this section we present our results as well as a comparative evaluation with baseline approaches. We experimented on 100 product links from two major online catalog aggregators, eBay and BestBuy. These links were equally distributed among a subset of available product categories.

Baselines *IE*: In this method we only utilize the IE phase of our system. The extracted keyphrases serve as ads. There is just one check regarding the overlap between different n-grams. In this premise, lower order n-grams that are in-

[2]`http://nlp.stanford.edu/software/tagger.shtml`

[3]`https://support.google.com/merchants/answer/1705911`

[4]`http://www.speech.sri.com/projects/srilm/`

Table 1: Examples of generated promotional text from all methods with top scores

Method	Product Name	Snippet
IE	Canon PIXMA iP100	*Auto Image Fix function automatically adjusts image*
IE+HG	Dell XPS Desktop	*Microsoft Windows 8 64-bit operating system preinstalled*
IE+SA	Samsung Galaxy S	*Cell Phone Free Shipping * No Contract Required * $25 Free Extras*
IE+HG+SA	Virgin Mobile - LG Optimus	*Bluetooth compatibility For wireless communication*
IE+NLG	VIZIO ESeries HDTV	***VIZIO ESeries with effective refresh rate, Low Price Guarantee***
IE+NLG+SA	Fujifilm Finepix JX580	***Artistically enliven photos, instantaneously increases shutter speed***
IE+CP	Dell Ultrabook	*Corning Gorilla glass ensures durability*

Table 2: Interjudge Agreement

Criterion	$P(D)$	$P(E)$	kappa statistic
Attractiveness	0.891	0.553	0.756
Clarity	0.912	0.561	0.800
Relevance	0.944	0.541	0.877

cluded in higher order n-grams are removed from the candidate list. $IE + HG$:We expand IE with a heuristic grammar check using Penn Treebank to eliminate candidates that a. contain conjuction, determiner, or comparative adverb in the last position of the sentence b. the tree confidence score of the Stanford lexicalized parser does not surpass a certain threshold ($\tau = -87$). $IE + SA$: Combination of IE along with sentiment analysis, by removing any retrieved phrases that are classified as negative. $IE + HG + SA$: Combination of IE with HG and SA .

Principal Method Variations Here, we introduce our integrated method. For every webpage in our dataset, we begin with stopword removal before proceeding to the IE phase. Additionally, we continue with the NLG phase ($IE+NLG$). Another variation is to extend $IE + NLG$ with the addition of the SA task, hence we remove any retrieved negative keyphrases from the IE phase before we proceed to the NLG phase ($IE + NLG + SA$). Additionally, we wanted to test the performance of a model that takes into account IE phase and proceeds with a very basic candidate pruning by using only the Information and Readability Ranking Functions ($IE + CP$).

Results The evaluation was held following a blind testing protocol by two groups of human evaluators, in total of 12 volunteer researchers. Each group was given the same set of {landing pages, product names, ad snippets} in order to measure the interjudge agreement between those two groups. For each method, three generated ads were presented to the evaluators. The evaluators were asked to answer if the ad snippet met the following criteria: a. Attractiveness A: Is the snippet attractive in order to prompt clicking on the ad? b. Clarity C: Combination of Grammaticality and Readability. Is the snippet structured in a comprehensible form? c. Relevance R: Is the snippet representative for the corresponding landing page? As an overall score we calculate the Harmonic Mean H, as we need an average of rates for the previous criteria. In Table 2 we present $P(D)$ (observed proportion of the times the judges agreed), $P(E)$ (proportion of the times they would be expected to agree by chance), and *kappa statistic* (correction of a simple agreement rate for the rate of chance agreement). In Table 3 we present the scores for each method. Some variations on baselines, which in first glance seemed that they would achieve better performance, actually went worse as they were eliminating candidates with very simple errors that seem not to influence significantly the evaluators.

Table 3: Criteria Rates Per Method

Method	A	C	R	H
IE	0.387	0.677	0.660	0.538
IE+HG	0.253	0.693	0.517	0.410
IE+SA	0.433	0.697	0.680	0.575
IE+HG+SA	0.293	0.647	0.550	0.443
IE+NLG	**0.527**	**0.854**	**0.943**	**0.726**
IE+NLG+SA	**0.593**	**0.850**	**0.937**	**0.763**
IE+CP	0.257	0.617	0.423	0.381

We performed the Wilcoxon rank-sum test in order to investigate whether the improvement between the most efficient methods and the best performed baselines is statistically significant. As a result, the differences in the A, C, R scores between IE & $IE + NLG$, IE & $IE + NLG + SA$, $IE + SA$ & $IE + NLG$, $IE + SA$ & $IE + NLG + SA$ were actually statistically significant ($p < 0.05$). Adding SA does not affect the computational cost while it improves each time the overall score, thus it provides an interesting solution for more prompting ads. In Table 1 we present examples of generated promotional text that met all the criteria.

5. CONCLUSION AND FUTURE WORK

We have proposed an engine that aims to help advertisers in creating promotional text for their online advertising campaigns in an automated and massive way by leveraging an Advertising Language Model and Sentiment Analysis. In future work we plan to recognize automatically more information about a product e.g. price and offers or discounts. Other interesting aspects that could be studied are the enrichment of the corpus for a more complete Language Model, the classification of products and services, and further evaluation using CTR from advertising campaigns.

6. ACKNOWLEDGMENTS

The research of S. Thomaidou is supported by the NSRF Program: Heracleitus II. Prof. M. Vazirgiannis is partially supported by the DIGITEO Chair grant LEVETONE in France. We would like to thank Prof. I. Androutsopoulos and M. Karkali for their helpful comments as well as our evaluators contributing to our experiments.

7. REFERENCES

[1] K. Bartz, C. Barr, and A. Aijaz. Natural language generation for sponsored-search advertisements. EC'08.

[2] A. Z. Broder, E. Gabrilovich, V. Josifovski, G. Mavromatis, and A. J. Smola. Bid generation for advanced match in sponsored search. WSDM'11.

[3] A. Fujita, K. Ikushima, S. Sato, R. Kamite, K. Ishiyama, and O. Tamachi. Automatic generation of listing ads by reusing promotional texts. ICEC '10.

[4] E. Gabrilovich. Ad retrieval systems *in vitro* and *in vivo*: Knowledge-based approaches to computational advertising. ECIR'11.

[5] K. Ganesan, C. Zhai, and E. Viegas. Micropinion generation: an unsupervised approach to generating ultra-concise summaries of opinions. WWW'12.

[6] S. Ravi, A. Z. Broder, E. Gabrilovich, V. Josifovski, S. Pandey, and B. Pang. Automatic generation of bid phrases for online advertising. WSDM'10.