The Workshops of the Thirty-Second
AAAI Conference on Artificial Intelligence

Thematic Distillation and Point of View Extraction for Enterprise-level Documents

Elham Khabiri, Wesley M. Gifford, Pietro Mazzoleni, Dharmesh Vadgama
IBM Research
1101 Kitchawan Rd.
Yorktown Heights, NY 10598

Abstract

An "elevator pitch" is a brief, persuasive speech that an experience seller can use to attain the attention of a prospective client. Unfortunately, when selling complex enterprise products and solutions, there is no one pitch that works for all customers. To craft a good pitch, a seller must study a large amount of documentation, including product descriptions, client references, and use cases. Leveraging experience developed over the years, sellers then determine which marketing message will work best with a client. The goal of our research is to automatically create knowledge snippets from a large set of enterprise documents that can be used in elevator pitches. We refer to these snippets of text as points of view (POVs). Our method is based on natural language understanding (NLU), clustering and ranking techniques where the most relevant and informative content are selected as POVs for a given product. In addition, our approach is tailored to create POVs for a given aspect of the product, like the business challenges or the benefits of deploying the product. In this paper, we present our initial results in analyzing thousands of client references and programmatically creating POVs for hundreds of IBM solutions. Our tool has been deployed and is being tested by a group of IBM sellers. While specifically built for IBM sellers and business partners, our solution has broad applicability in the delivery of marketing messages for complex enterprise solutions.

1 Introduction

Business to business (B2B) companies, especially ones working with medium-to-large enterprises, still rely heavily on in-person communication to drive marketing and sales. When a sales person begins the conversation with a prospective client, a critical task is crafting a personalized marketing message which would resonate with the client. We refer to such personalized marketing messages as an "elevator pitch". In other words, an "elevator pitch" is a brief, persuasive speech that an experienced seller can use to attain the attention of a prospective client. Typical components of an effective elevator pitch comprise answers to questions like "why is this product relevant to my organization", "where have you deployed it before", or "which business benefits will I obtain from using it". Unfortunately, creating an "elevator pitch" is complex as it can vary by customer or seller selling style, and it has to constantly evolve along with the product. To craft a good pitch, a seller must study a large amount of documentation, including product descriptions, client references, and use cases. Leveraging experience developed over the years, sellers will then learn the most effective marketing messages for a client. Unfortunately, most sellers do not have the experience, capacity, or time to distill such a large amount of data to create personalized marketing messages. This creates sub-optimal communication with the client and has a direct impact on revenue.

The goal of our research is to help sellers in creating effective "elevator pitches". Note that our objective is not helping sellers find the right document, but rather to automatically create knowledge snippets to be used in an elevator pitch from a large collection (possibly hundreds of thousands) of documents. We refer to these knowledge snippets as points of view (POVs). Operating in the broad research area of knowledge summarization, our method is based on natural language understanding (NLU), clustering and ranking techniques where the most relevant and informative content are selected as POVs for a given product. A key capability of our approach is the ability create POVs for a given aspect of the product, like business challenges or business benefits in adopting the product, or competitor-analysis insights.

In this paper we present our initial results in analyzing thousands of client references and programmatically creating POVs for hundreds of IBM products and solutions. Our tool has been deployed and is currently being evaluated by a group of IBM sellers. While specifically built for IBM sellers and business partners, our solution has broad applicability in the delivery of marketing messages for complex enterprise solutions.

The rest of the paper is structured as follows. In section 2, we compare our method with the existing state-of-the-art algorithms. Section 3 explains the motivational use-case as well as the data we used for our analysis. In section 4, we describe the end-to-end procedure to extract POVs. In section 5, we explain different modeling approaches and how we use those with our data. In section 6 we briefly describe the tool we created for IBM sellers and initial user feedback. Section 7 concludes the paper and discusses future efforts.

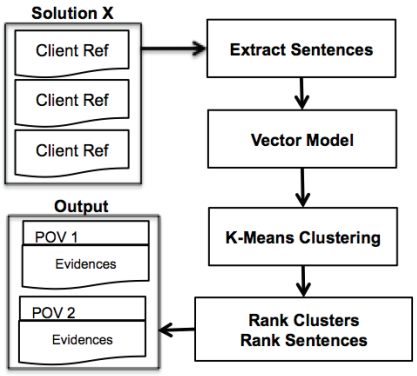

Figure 1: POV extraction consists of clustering similar sentences, ranking sentences, ranking clusters, and linking sentences to corresponding documents.

2 Related Work

This work is by its nature closely related to the general topic of summarization. Although our application is specific to an enterprise, the objective is identical to many summarization tasks; to produce a dense, informative summary which we refer to as point of view (POV). There are many works in the area of extractive and abstractive summarization. Extractive models generate summaries by extracting important sentences from the original text and combining them to form a summary. On the other hand, abstractive models create summaries from scratch without being constrained to reuse phrases from the original text. Abstractive summarization, e.g. (Ganesan, Zhai, and Han 2010) and (Ganesan, Zhai, and Viegas 2012) is less common than extractive approaches and more difficult since significant amounts of training data are required. Thus, most research on summarization focuses on extractive methods. Many of the extractive methods are based on graph algorithms, including (Khabiri, Caverlee, and Hsu 2011), TextRank (Mihalcea 2005) and LexRank (Erkan and Radev 2004) for document summarization. Similar to Google's PageRank algorithm (Page et al. 1999) or Kleinberg's HITS algorithm (Kleinberg 1999), these methods first build a graph based on the similarity relationships among the sentences in a document. Then the importance of a sentence is determined by taking into account the global information on the graph recursively.

Recently, there have been several papers on using neural networks to generate abstractive summaries. (Rush, Chopra, and Weston 2015) adapted the feed-forward neural network language model which was originally described by (Bengio et al. 2003). They applied a local attention-based model so that each word of the summary is generated when conditioned on the input sentence. Also, (Chopra, Auli, and Rush 2016) proposed a conditional recurrent neural network which acts as a decoder to generate the summary of an input sentence. Both of the above works are inspired by the proposed architectures for machine translation (Bahdanau, Cho, and Bengio 2014). They have extended an encoder-decoder architecture, allowing a model to automatically search for parts of a source sentence that are relevant to predicting a target word. More recently, (See, Liu, and Manning 2017) proposed a pointer generator model with coverage mechanism that is built on top of a sequence-to-sequence attention model (Nallapati et al. 2016). It is shown to prevent undesirable behaviors such as inaccurate factual details, inability to deal with out of vocabulary words, and provides summaries with non-repeating concepts.

The above works have used large input-summary pairs to train their models. Unfortunately for specific domains such as marketing, there are no standard training sets, making development and application of abstractive models difficult. Therefore, in this paper we explore different techniques of extractive summarization. We have leveraged some of the neural network based techniques for word embeddings as were suggested in (Mikolov et al. 2013b) and (Le and Mikolov 2014).

3 Motivating Scenario and Data

The dataset we used for our work consisted of 115 IBM industry solutions and approximately 10,000 client reference and case study documents.

IBM industry solutions represent IBM offerings specifically tailored to one industry, such as healthcare, retail or financial services. As an example, the "Health system performance & optimization" solution for the healthcare industry aims to optimize business operations to meet changing industry demands. Creating POVs for industry solutions is particularly challenging for sellers as they need to include both business and technical aspects of the offering. For each of the 115 industry solutions, we used the title and description to semantically categorize the offering.

The body of knowledge used to extract POVs consists of approximately 10,000 client reference and case studies. Client references and case studies describe success stories of how clients teamed up with IBM to address specific challenges. The content includes fields such as the client name, client description, business needs, solution benefits, and industry. The business needs and solution benefits fields are free text fields, and contain a few sentences up to several paragraphs of content. Note that these documents do not include a specific tag for industry solution as the success stories might have elements which are relevant to multiple solutions. Determining which client references are relevant to each industry solution was one of the challenges we had to address.

4 Extracting POVs

The process of extracting POVs from a large set of documents is depicted in Fig. 1. It begins with mapping the solutions to documents, since there is no explicit mapping available. Second, all the documents pertaining to each solution are integrated into a group for further processing. For these groups of documents the business needs and solution benefits sections are kept separate. Then, from each group, similar sentences are clustered together based on their vector representations. The ranking of the clusters, as well as the sentences within each cluster, are done according to sev-

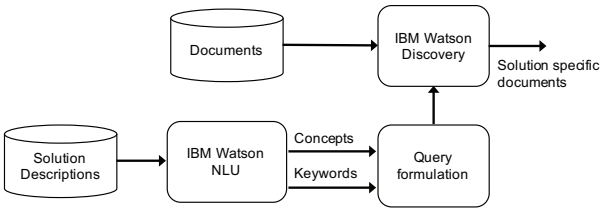

Figure 2: Process to map solutions to documents

eral metrics described later in this section. Finally, the POVs are extracted from the sentences with the highest combined score in each cluster. Many of the above steps are based on the vector representation of the sentences. Approaches to creating the vector model are described in Sec. 5.

Mapping Solutions to Documents

Frequently in enterprise environments documents will not be properly tagged, or will be tagged in a different way than is needed. In our case we first need to associate the source documents with solutions. While source documents are tagged with some product names, these names have evolved over time and hence they do not exactly match our solution names.

Our approach to handle this is to determine which source documents are most relevant to a particular solution by considering the concepts and keywords expressed in both sets of documents. We begin this process by extracting concepts and keywords from solutions using IBM Watson Natural Language Understanding[1]. This tool has the capability to analyze text in a variety of ways, including extracting concepts, entities, keywords, sentiment, etc. We found that concepts alone did not capture significant information about the solutions given the typically short length of their descriptions, but adding the keywords improved the quality of the results.

Once we have a set of concepts and keywords we need to identify the most relevant source documents. To facilitate searching over documents and extracting keywords and concepts en masse, we loaded the source documents into IBM Watson Discovery[2]. Now, we use a query constructed from the top keywords and concepts of each solution description to return a ranked list of matching documents. This process is carried out across all solutions, saving the top 200 documents that resulted from the concept and keyword based query. The process of mapping solutions to documents is depicted in Fig. 2.

Identify Target Sentences

For our use case, we need to focus on sentences that pertain to the business needs and solution benefits for a particular solution. To this end, we perform a preprocessing step to retain only the sentences specifically addressing needs and benefits for use in the final summary. However, in training our models, we are not constrained to such sentences. This step is only needed when choosing the POV sentences.

For all the sentences in the needs and benefits sections, the subject-verb-objects (SVOs) and later the subject-verb-adjective-objects (SVJOs) with their occurrence frequencies are extracted. As an example, we see that a SVO such as "bank-gain-platform" and its related SVJO "bank-gain-scalable-platform" has a relatively high frequency. This serves as a signal that the verb "gain" and its synonyms should be among the terms that reflect the needs and benefits.

In order to extract the synonyms of these terms, we use a word2vec model that is trained on GoogleNews (Mikolov et al. 2013a) (Mikolov et al. 2013b). As a result verbs including "necessitate", "demand", "support", "facilitate" and "accommodate" are considered as relevant terms that reflect a need or benefit concept. After filtering, we only consider the sentences with such verbs as candidates for POVs.

Clustering

In order to group the references with the same objective into a common POV, a clustering algorithm is applied on all the needs and benefits across all the documents associated with a single solution. We refer to the most representative sentence of a cluster as the POV and the rest of the sentences in the same cluster as the "evidence" of that POV. We want our algorithms to produce sentences that are distinguishable and represent distinct knowledge that should be consumed by the sellers. The primary purpose of POV extraction is to identify the best content that reflects the diverse set of benefits and needs for each solution, while eliminating conceptually similar pieces of content. From the previous mapping step this part of the process receives all the benefits and needs sentences from the selected subset of documents which are most relevant to the target solution. In order to cluster similar sentences, we use k-means clustering, a standard method to group relevant content. Here each sentence is converted to its vector representation with size M. We discuss different ways of modeling the vector representations for each sentence in Sec. 5. Given the number of clusters and a set of sentences as input, the k-means algorithm assigns each sentence to the cluster whose center is the nearest. The cluster center is the average of all the vector representations of the sentences in that cluster.

One challenge is to identify the right number of clusters for each solution. This is a parameter that should be determined beforehand. One way to estimate the number of clusters is to use silhouette analysis (Rousseeuw 1987) which reflects how similar an object is to its own cluster (cohesion) compared to other clusters (separation). To perform the analysis, for each candidate number of clusters, k_i, the average silhouette score of all the sentences in all the clusters are calculated. The k_i with highest silhouette score gets selected as the parameter value.

Using silhouette analysis comes with challenges. In our case, some clusters for solutions contain similar sentences about the same repeating themes, while others contain very diverse sets of content. The silhouette score appears to be low for all candidate values of the number of clusters. In

[1]https://www.ibm.com/watson/services/natural-language-understanding/

[2]https://www.ibm.com/watson/services/discovery/

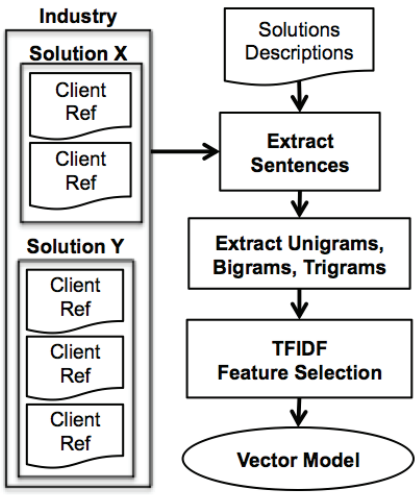

Figure 3: A vector model is used for clustering similar sentences, measuring similarity between each sentence and the solution description, measuring similarity between each sentence to the centroid of its cluster, and measuring the importance of each sentence, using TF-IDF.

such cases we rely on a fixed number of clusters that we determined beforehand, which by experimentation is the total number of sentences divided by 15.

Ranking

Two types of ranking take place in the POV extraction process. One is ranking the clusters to find the best POVs for a given solution, and the other is ranking the sentences within each cluster to find the best sentence to represent that cluster.

Ranking Clusters Cluster ranking is used to find the most relevant POVs for a given solution. There are several potential metrics which can be used to rank the clusters, including cosine similarity, term frequency-inverse document frequency (TF-IDF) related metrics, cluster size, and cluster silhouette. We performed manual investigation of these metrics to determine which is most effective. We found that cosine similarity of the centroid of the cluster with the solution description provided the most meaningful ranking. The rationale behind the centroid method is as follows. First, the centroid of a cluster should contain information that is close or central among the sentences in that cluster. Similarly, the description of the related solution contains terms specific to the solution. Thus, the similarity between the two should reveal a more relevant cluster. The cluster score is defined as follows:

$$cluster\_score = \frac{cluster\_centroid \cdot solution\_desc}{\|cluster\_centroid\|_2 \|solution\_desc\|_2} \quad (1)$$

where $cluster\_centroid$ and $solution\_desc$ are the vector representations of the cluster centroid and solution description, respectively.

Ranking Sentences Now that we have the clusters ranked, we are interested in ranking the sentences within each cluster. This is done to elect the best, most relevant sentence to serve as POV and represent the information in the cluster. The first approach to ranking the sentences in a cluster is by awarding more points to sentences containing "important" terms. The intuition is that sentences containing more significant terms are themselves more significant. This metric is explained more in detail in the Sec. 5. Another metric is the similarity between a sentence and the centroid of the cluster containing that sentence. Yet another metric is the similarity of the sentence to the solution description.

The main challenge is to figure out which metric is best among these possible options. Identifying the effectiveness of each metric is subjective and needs careful manual evaluation. Based on the intuition behind the metrics we selected the weighted average of the following two metrics: normalized average of the TF-IDF scores of the terms in a sentence and the normalized similarity of the sentence to the centroid of its cluster. Therefore the sentence score is given by:

$$sent\_score = \frac{norm\_tfidf(sent) + norm\_sim\_centroid(sent)}{2} \quad (2)$$

where

$$norm\_tfidf(sent) = \frac{\sum_{t_i \in sent} tfidf(t_i)/len(sent)}{max\_tfidf} \quad (3)$$

and

$$norm\_sim\_centroid(sent) = \frac{sim\_centroid(sent)}{max\_sim\_centroid} \quad (4)$$

The normalization plays an important role to bring different metrics such as $tfidf$ and $sim\_centroid$ into comparable ranges. Since they are all considered relative to their maximum values in their containing solution, the averaging operation of the metrics becomes possible. The max_tfidf is the maximum TF-IDF score of the all the terms which appear in a solution. The $max\_sim\_centroid$ is the highest similarity between a sentence and the solution description among all the sentences in a solution.

5 Modeling

We use two types of modeling algorithms to create vector representations for our input sentences. One is based on a TF-IDF model with size equal to the size of the dictionary. The other is based on the skip-gram model with 300-600 dimensions.

TF-IDF Model

The TF-IDF model is based on numerical statistics that reflect how important a word is to a document in a corpus (Leskovec, Rajaraman, and Ullman 2014). Each term and consequently each sentence is mapped to a vector with the size of the corpus dictionary. Each element of this vector is the TF-IDF score of the term used in the sentence. Our dictionary size is 200,000 terms including unigrams, bigrams and trigrams. The importance of each sentence is the average of the TF-IDF of terms used in that sentence.

Figure 4: Screenshot of the seller expertise advisor tool

Feature Selection The process of selecting a subset of the terms occurring in the training set and using only this subset as features in text classification is called feature selection. Here the main purpose behind feature selection is to decrease the size of the effective vocabulary and choose terms that are more discriminative among solutions of an industry. In our case, as a result of feature selection, the feature size is reduced from 200K to 20K.

As a reminder, our goal is to extract POVs for each solution in an industry. Feature selection results in identifying terms and phrases that are important for distinguishing one solution from the rest of the solutions so that the POVs reflect unique concepts to that solution. Sometimes though, there are many similarities among the solutions themselves. For example, "Customer and Risk Insight" vs "Customer and Risk Engagement" are two separate solutions, but have many similar terms. For these cases even feature selection may not be useful. We need to rely on other metrics such as similarity of a sentence to its description in the sentence ranking process.

Skip-gram Modeling

A disadvantage of TF-IDF model is that it is unable to capture the similarities between the words. We use the skip-gram model proposed by (Mikolov et al. 2013a) and trained it on 115 Industry Solutions, 448 Marketplace Solutions and 168 Software Solutions to obtain domain-specific, 300-dimensional word embeddings. A sentence is then represented by the mean of its word embeddings.

In general, larger corpora yield more accurate word embeddings. It is also beneficial to leverage the pre-trained Google News model (Mikolov et al. 2013a) with 3 million 300-dimensional English word vectors that is publicly available[3]. Using this pre-trained model alone will not be sufficient, since it lacks many of the domain specific words for our application.

We propose a method that concatenates the two 300-dimensional vectors. One is the vector resulting from domain specific training, while the other is from the pre-trained model. In the case where a word is not found in one model, the other model can provide the word representation.

One additional improvement is to weight the vector representation with the TF-IDF score of the terms. In this way, we consider the importance of a term other than the word embedding that reflects the relevance between the similar terms. The weighting is done as follows:

$$vect(sent) = \sum_{t_i \in sent} vect(t_i) * tfidf(t_i) \qquad (5)$$

where $vect(t_i)$ is the embedding for term t_i, the output of the skip-gram model that is trained on both the global and domain specific corpora.

6 Seller Expertise Advisor Tool

As part of our work, we created both Rest APIs and a tool, called "Seller Expertise Advisor" (SEA) for sellers to access our dynamically generated POVs for all 115 industry solutions. IBM sellers use SEA when they need to prepare for a customer meeting or they need to get up to speed (or refresh their knowledge) on a given offering.

[3]https://code.google.com/archive/p/word2vec/

A sample screenshot of our tool in presented in Fig. 4[4]. The figure shows three sample POVs created for "Customer Insights", an IBM industry solution for the financial sector. Sellers can drill down into each POV and access all the documents, referred to as evidence, which are relevant to each POV. For each piece of evidence, SEA highlights the part of the document semantically relevant to the POV. The original client reference or case study is also accessible from the tool. In a sense, SEA recommends the marketing message for the solution and gives sellers the opportunity to explore content and learn more about such POV. In the tool, sellers can provide feedback by voting if they like or dislike the POVs. We use such information to continually improve the quality of the results.

The tool has been deployed to thousands of IBM sellers and initial feedback has been extremely positive. Sellers are excited about the solution as it drastically reduces the time needed to prepare for a meeting and helps make them more effective in learning about a product.

7 Conclusion

In this paper we proposed and applied information retrieval and machine learning based techniques to extract POVs from a large collection of documents. The solution allows identification of key snippets of information which we believe help facilitate positioning of solutions by sellers as they engage with clients. This approach will help less experienced sellers more quickly gain the knowledge necessary to be successful.

We have currently deployed our initial solution in an enterprise environment for 115 solutions across 17 industries. As there is currently no ground truth to evaluate the quality of the POVs, and hence the overall performance of the solution, we have instrumented our deployment to capture feedback from sellers. We envision using this feedback to better evaluate the performance of the various design choices described in this paper, as well as improve the algorithms in the future. Our future efforts are focused on moving towards abstractive summarization by applying neural sequence-to-sequence models to our domain-specific corpus. We are also interested in considering options to create suitable training datasets.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Chopra, S.; Auli, M.; and Rush, A. M. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 93–98.

Erkan, G., and Radev, D. R. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res. (JAIR)* 22:457–479.

Ganesan, K.; Zhai, C.; and Han, J. 2010. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, 340–348. Association for Computational Linguistics.

Ganesan, K.; Zhai, C.; and Viegas, E. 2012. Micropinion generation: An unsupervised approach to generating ultra-concise summaries of opinions. In *Proceedings of the 21st International Conference on World Wide Web*, 869–878. ACM.

Khabiri, E.; Caverlee, J.; and Hsu, C.-F. 2011. Summarizing user-contributed comments. In *Proceedings of the 2011 International Conference on Web and Social Media*.

Kleinberg, J. M. 1999. Hubs, authorities, and communities. *ACM Computing Surveys (CSUR)* 31(4es):5.

Le, Q., and Mikolov, T. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 1188–1196.

Leskovec, J.; Rajaraman, A.; and Ullman, J. D. 2014. *Mining of massive datasets*. Cambridge University Press.

Mihalcea, R. 2005. Language independent extractive summarization. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, 49–52. Association for Computational Linguistics.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 3111–3119.

Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B.; et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: bringing order to the web. Technical report, Stanford InfoLab.

Rousseeuw, P. J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 20:53–65.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

[4]Note that black boxes have been applied to the figure to hide confidential information.