CS474 term project paper

Jihee Park
KAIST
Daejeon, Korea
j31d0@kaist.ac.kr

Junseop Ji
KAIST
Daejeon, Korea
gaon0403@kaist.ac.kr

Soyoung Yoon
KAIST
Daejeon, Korea
lovelife@kaist.ac.kr

ABSTRACT

Due to the massive increase of news articles in the internet, the importance of topic analysis and issue tracking is growing. However, the massive amount of data makes people hard to do the work manually, so the automatic process held by the machine is needed. In this paper, we suggest an automatic news analysis process, which consists three steps: *1. trend analysis*, *2. on-issue event tracking*, and *3. off-issue event tracking*. After the experiment, we see that our trend analysis model clusters the news articles by topics very well, and event tracking models find out the events for each issue(topic).

KEYWORDS

topic analysis, event tracking, news

ACM Reference Format:
Jihee Park, Junseop Ji, and Soyoung Yoon. 2019. CS474 term project paper. In *CS474 Term project report* . KAIST, **??** pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Online contents has grown big recently, and people are viewing more news on the internet. However, due to the heavy amount of data(news), it has a limitation to categorize the news manually. Also, it is almost impossible to track the events related to the issue while looking all the articles.

Looking at all yearly issues, we can see that the news articles can be clusted into some issues. For example, in the case of the year 2017, there was a lot of news related with the former president and her political crimes. The term *Trend Analysis* means clustering those kind of news articles and analyze the clusters. And, for each issue, we can see the *events* related to the issue and we can make the timeline of the main events for an issue. In detail, there are two kinds of event: the first one is the event which is directly related to an issue, and the second one is not directly linked, but topically related issue. In the paper, we call the first one as *"on-issue event"*, and the second one as *"off-issue event(related-issue event)"*

In this paper, we suggest a method to analyze the trends and track the events by three steps: 1. *Trend Analysis*, 2. *On-issue Event Tracking*, and 3. *Off-issue Event Tracking*.

CS474 Term Project, 2019 Fall, KAIST

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

After doing all the progress, we could analyze the yearly trends of Korea from 2015 to 2017. Also, we tried to track down some important issues extracted from above. We captured four quarterly events for each issue, and also tracked several off-issue events.

Our research is important and effective because we minimized the human(manual) efforts throughout the progress, for summarizing the 270K news articles. The methods we suggest in this paper can be widely used in yearly trend analysis, not just news, but marketing, research, or the other fields as well.

2 TREND ANALYSIS

2.1 Data Preprocessing

2.1.1 Data Format. As described in the READ.ME of data provided, The targeted data is from the Korean Herald, National Section news. The period of the dataset is from 2015 to 2017. The Crawled date of the dataset is 2018-10-26. Data format is Json, and there are total of 6 data headers - title, author, time, description, body, and section. Total of 23769 news are included in this dataset.

2.1.2 Load Data. In order to load the data, the instructions recommended at READ.ME are followed. Pandas library is used for better storing and access of the news text.

2.1.3 Libraries Used. For this project, we used pandas, gensim, nltk, and neuroner python libraries. The install requirements are found on install.sh of the github repository.

2.2 Previous approaches

Issue trend analysis can be seen as a part of Topic modeling. By searching fields of recent Topic modeling, LDA has shown to have good performance. As a result, LDA is used as a baseline algoritm for this project. A recent study(2018) on Topic Modeling shows that Topic Quality improves when Named Entities are promoted.[**?**] This paper proposes 2 techniques: 1. Independent Named Entity Promoting and 2. Document Dependent Named Entity Promoting. Independent Named Entity Promoting promotes the importance of the named entities by applying scalar multiplicaion alpha to the importance of the named entity word. Document Dependent Named Entity Promoting promotes the importance of the named entities by setting the weights of the named entities as maximum term-frequency per document. For Independent Named Entity Promoting, the value of alpha can be changed flexibily, but results conducted by this paper shows that setting alpha as 10 showed the best results. We take advantage of this paper's idea on Independent Named

2.3 Experiments

2.3.1 Data Tokenization. Several attempts were taken before we finalize the way Tokenization was done. Lemmatization is not always good At first try, Lemmatization(converting words into base forms) and removal of stopwords were conducted before we run the LDA algorithm and extract Named Entities. We thought that converting words into base forms and reducing the total vocabulary size would increase the performance of topic modeling. Stopwords were taken from `res.append(lemmatize(raw\_text, stopwords=stopwords))`, and lemmatization functon was taken from `word_tokenize from nltk.tokenize`. `ne_chunk(pos_tag(preprocessed_text), binary=True)`. But after we do lemmatization, remove stopwords, and tokenize the data, no Named Entities were extracted from the preprocessed corpus. We think the reason for this is as follows. First, words are all converted into lower case when we do lemmatization. This makes the Named Entitiy Recognition system(NER system) to work poorly because we have removed the original information(Upper case information), and word that starts with an upper case has a high probability that it is a "Proper pronoun", or "Unique word". We lose this sign of information. Second, words are transformed into their base forms, limiting NER system to detect specific words. There also could be cases that the words are transformed into meanings other then their original meanings. For example, "Cooking" and "Cooker" are both converted into "cook" when they are lemmatized, and this makes the word to lose the original information. Third, original relationships between words are lost, because of the removal of stopwords. When we do NER, we have to do the POS tagging of the sentence and then input both the word sequence and the POS sequence of the text. But when we artificially remove stopwords and then do NER, original relationships between words are disrupted and broken. This limits NER system to perform well.

For these 3 reasons, we decided to NOT apply lemmatization for tokenization, because lemmatization lose so much information about the original text and disrupts the NER system's ability to detect Named Entities properly. We decided to just do `" \xec ' ( ) . ,`, do POS tagging and then do NER.

2.3.2 Extract NER. By using ne_chunk from nltk and pos_tag from nltk.tag, we extracted Named entities from the original news dataset. NER also extracts multi-word information of Named Entities other than just classifying whether a word is a named entity or not, so we decided to use that information. We store single-word Named Entities and multi-word named entities separately. As a result, NER and multi-word extraction of NER are both processed.

Below figure is the topic modeling result(of all time lengths from 2015 to 2017) WITH NER Promoting and WITHOUT NER Promoting. We can see the difference between those two results, and we can conclude topic modeling with NER promoting shows better performance.

2.3.3 Improvement - apply neuroNER instead of nltk's Named Entity Recognition. The topic modeling paper that we referenced used neuroNER for Named Entity Recognition. NeuronNER is an easy-to-use program for named entity recognition based on neural networks presented in emnlp 2017. [?] This neuroNER tool is trained on CONLL2003 dataset and recognizes four types of NE: person, location, organization and miscellaneous. NeuroNER also extracts multi-word information, so we use this multi-word information just as the previous NER did. Instead of using `ne_chunk(pos_tag(preprocessed_text), b`, we change NER extraction to use

```
nn = neuromodel.NeuroNER(train_model=False, use_pretrained_model=T
nn.predict(preprocessed_text)
```

to extract Named Entities from the text.

2.3.4 Run LDA with neuroNER promoting. First, we split the dataset each year. Then, get tokens for each document with promoted NER frequency (X 10). With this corpus, run the LdaModel with num_topics of 10 and num_words of 30 to 50. At first try, we directly ran LDA on NER boosted news dataset. but with this approach, we found out that stopwords are classified as top(important)words according to the result of LDA. So we decided to remove stopwords after all the preprocssing(including NER weight promoting)are done. The timing of removal of stopwords are important, as removing stopwords before NER will affect the NER result(Removal of stopwords before POS Tagging will affect the POS Tagging result). Stopword removing are done right before feeding the tokens into LDA. After the removal of stopwords, we could see that the results were much better.

2.3.5 Tuning LDA hyperparameters. We set num_topics to 10 for LDA becuase we need to extract top 10 important issues from each year. At first, we decided to train the LDA model with num_topics of 10 and num_words of 15. But the results were not very explainable. Also, the only removed word was the stopword after tokenization. Therefore non-ascii character, or unrelated words such as `" \xec ' ( ) . ,` were introduced in the topic result. To extract useful information, we removed those unuseful information and increased num_words for each topics to 50 to see more related words including each topic. First we set chunk_size to 2000, num_iterations to 4000, and alpha to 'auto'. We changed chunk_size to 4000 and increased num_iterations, and see if the result improved. But there was no significant change on the results. We finally decided to set num_topics to 10, chunk_size to 4000, iterations to 500, and passes to 30.

2.3.6 TroubleShooting. In order to increase the performance of Topic modeling, various approaches were taken. The first trial was to divide news dataset into given sections then do LDA modeling for each year, for each topic. But this approach can not detect the top 10 trending issues. Increasing the total topic size to more than 10 makes us difficult to analyze which topic is the top 10 most trending topic. This happened to be the same problem when we increase the total topic_size to more than 10. But, setting the total topic_size to 10 also has problems. By setting the total topic_size to 10 for each year, many topics can be concatenated into one. For this case, we filter out the majority topic by looking at the extracted tokens for each topic. Also, for the herald dataset, words related for "Korea" (Korean, North Korea, South Korea, Korean, ...) are used very frequently accross all topics, so the Topic analysis result for this also showed great frequency of words related to "Korea", making "Korea" unuseful for topic detection. Also there were topic clusters that were hard to analyze the keyword. Also, NER are good at extracting multi-words, but was not good at triple or more words.

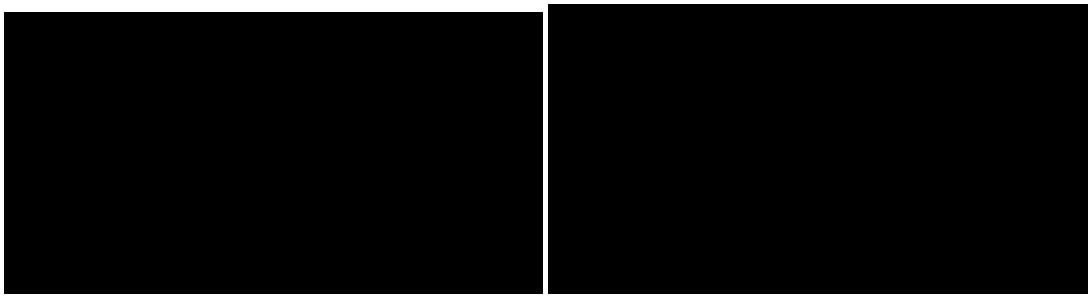

Figure 1: Topic modeling result before/after NER

For example, the neuroNER output of "Moon Jae-in eat food" was "Moon Jae", not 'Moon Jae-in". We could see the inherent limitations of extracting 2 or more words just by using NER. To overcome those problems, more data preprocessing should be done. For example, multi-word extraction could be done to increase performance, and If we link the corresponding news article that best represents a particular topic, it will make Topic modeling more analyzable.

3 ON-ISSUE TRACKING

For on-issue tracking, we first divide news articles quarterly. Then we classify news articles in each quarter group into 20 issue categories. For each classified group, each articleâĂŹs 5W1H(when, where, who, what, why, how) is extracted and counted. The most frequent 5W1H will represent an on-issue event for the quarter.

Figure **??** shows the structure of the on-issue tracking process.

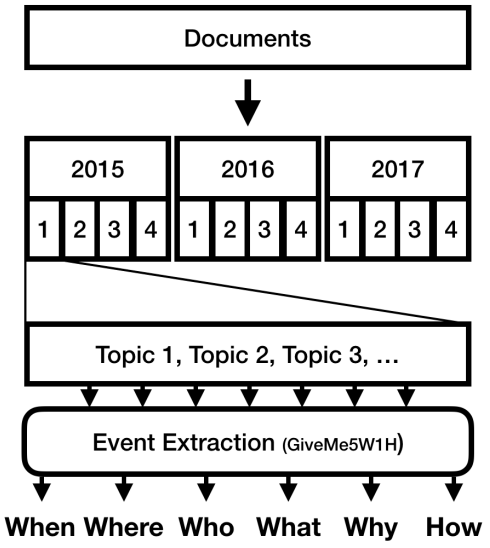

Figure 2: A brief diagram of on-issue tracking process.

3.1 Quarterly Division

We divided all news articles quarterly. The groups contain news articles those are written in *2015 Q1, 2015 Q2, âĂę, 2017 Q4, 2018 Q1.* *2018 Q1* group contains only articles written in January, 2018, so we merge the last group with the group *2017 Q4*. The reason why we divided the data quarterly is, the quarter is a semi-standard in the field of yearly statistics. If we divide yearly, there will be only three groups and it will not have high accuracy if we make a timeline of the events. So we chose a quarterly division to make reasonable results.

3.2 Articles in the Quarters Categorization

With LDA model we have trained at trend analysis project, we classify the documents in the quarter groups. If we give a tokenized sentence to the LDA model, the model outputs the probability for each group. We choose the group with maximum value, and assign the document to the group. So, for each quarter, there are 21 classified groups of news articles.

3.3 Event Extraction

For each group we divided from above, we extract the events with the approach of word frequency. For this step, we use a Python library called "giveme5W1H". The library is the state-of-the-art tool for extracting *when/where/who/what/why/how* features from the document. The library uses Stanford's CoreNLP library as its basic structure, and give analysis results when we give a title, lead, text, and a published date. We decided to use columns *title*, *description*, *body*, and a *time* from the given dataset as an input to get a result. For each group, we count the frequencies of each feature of the articles, and select the most frequent terms for each feature, treat them as an event.

In details, to reduce the time of event extraction, we choose two yearly issues from the list, and do event extraction for each issue. So each yearly issue has four event extraction result: in Q1, Q2, Q3, and Q4. For each quarter result, we identify an event based on the result and align them on the timeline.

Table ? is an on-issue tracking of the issue MERS. The events timeline of the issue MERS in 2015 is: **Korean aid worker sent(Q1) -> Korean government's MERS announcing -> Korean government reports no additional MERS cases -> Government created free medical school** .

4 OFF-ISSUE TRACKING

For off-issue tracking, we first categorize topics given as Trend analysis part. In this section, we denote a document as sequence of

tokens plus its created time $\mathbb{D} := (\Sigma^+, t)$, when $t \in \mathbb{R}$ (timestamp of creation time). and the set of document of topic a as $\mathbb{T}_a \in \mathcal{P}(\mathbb{D})$.

4.1 BoW extractuion

In first, we have to extract document in some space which we can analyze quantatively. We use BoW as morphism from document space to vector space $\mathbb{R}^N$, which we can analyze similarity of document. In addition, we add one more dimension to give information of document creation time. From pre-calculated set of tokens $\Sigma := \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, our transformation $b : \mathbb{D} \to \mathbb{R}^{n+1}$ is defined inductively as

$$\begin{cases} b([], t) := t * e_{n+1} \\ b(\sigma_i :: tl, t) := e_i + b(tl, t) \end{cases}$$

Then, morphism from $\mathbb{T}_a \in \mathcal{P}(\mathbb{D})$ to $\mathcal{P}(\mathbb{R}^{n+1})$ is naturally induced from b as $\phi(\mathbb{T}_a) = \{b(d) | d \in \mathbb{T}_a\}$

4.2 Relation between semantic of document and BoW

We know that there are documents and events which have similar meaning, but we cannot formalize it because we currently do not have model of language interpretation in metric space. But we can assume *such* space exists, i.e. there is an isomorphism $\phi : \mathbb{D} \to \mathbb{D}^\#$, when $(D^\#, d^\#)$ is metric space. It is not hard to assume this structure, since similar concept is already introduced as Entity comparison/Behavior comparison operator of Semantic algebra [**?**].

Our desired result is that b with euclidean distance successfuly models $(D^\#, d^\#)$, but we cannot show it because we do not have constructive definition of $D^\#$. But if it has sufficient approximation, (bounded approximation) We can derive more interesting properties (such as bounded error from BoW to Event space, etc).

Definition 4.1. b has approximation of ϕ with bound K, ϵ iff there exists an Lipshitz continuous π with K that $d^\#(\pi(b(d)), \phi(d)) \leq \epsilon$.

4.3 Relation between semantic of event and BoW

Once semantic of document is defined, we can build similar notion of event as metric space. To build such space, we first understand about relation between document and event.

- similar document refer similar event.
- similar event (even same event) may be refered by documents with far distance, but it is not arbitrarly far.

we can formulize this as logical formlua, with definition of $e : D^\# \to E^\#$. $((E^\#, e^\#)$ is metric space for event)

- if $d^\#(d_1, d_2)$ is sufficiently small, then $e^\#(e(d_1), e(d_2))$ is sufficiently small.
- when $e^\#(e(d_1), e(d_2))$ is small, it doesn't mean $d^\#(d_1, d_2)$ is small but is bounded.

begin with this fact, we can find very interesting property which generalize this: continuity.

Definition 4.2. e is Lipschitz continuous with K if and only if $e^\#(e(d_1), e(d_2)) \leq K d^\#(d_1, d_2)$.

We can check that if e is Lipschitz continuous with K_e, then above two property is satisfied. Also, it derives important fact: If we have approximation of semantics with bounded error, then there also exists approximation of event with bounded error.

Theorem 4.3. *b has approximation of ϕ with bound K, ϵ, then there exists $\pi_e : \mathbb{R}^{n+1} \to E^\#$ s.t. $e^\#(\pi_e(b(d)), e(\phi(d))) \leq K_e \cdot \epsilon$. (it means b has approximation of $e \cdot \phi$ with bound $K, K_e \cdot \epsilon$)*

Although proof is directly derived from Lipschitz continuity, it emphasizes that if we have bounded approximation of document, then it guarantees bounded approximation of event.

4.4 Event clustering

In this assumption about semantic of document ans event, we can build event clustering method. Before using techniques in R^{n+1}, we focus on how this clustering in R^{n+1} effects in $E^\#$.

Theorem 4.4. *if b has approximation of $e \cdot \phi$ with bound K, ϵ, then $e^\#(e \cdot \phi(d_1), e \cdot \phi(d_2)) \leq 2 \cdot \epsilon + K||b(d_1) - b(d_2)||$.*

Proof.

$$\begin{aligned} & e^\#(e \cdot \phi(d_1), e \cdot \phi(d_2)) \leq e^\#(e \cdot \phi(d_1), \pi_e(b(d_1))) + \\ & e^\#(\pi_e(b(d_1)), \pi_e(b(d_2))) + e^\#(\pi_e(b(d_2)), e \cdot \phi(d_2)) \leq \\ & \epsilon + e^\#(\pi_e(b(d_1)), \pi_e(b(d_2))) + \epsilon \leq \\ & 2 \cdot \epsilon + K||b(d_1) - b(d_2)||. \end{aligned}$$

□

It shows that, if we make good Vector transformation b, then it automatically guarantees bounded error for distance of extracted event, without construction of π, ϕ, e or any other. Begin with this fact, we derive constructive definition of partition for documents using approximated transformation b. To do that, we first define similarity relation for two documents.

Definition 4.5 (Similarity relation). $\approx_{\mathbb{R}^{n+1}, \delta} \in \mathcal{P}(\mathbb{D} \times \mathbb{D})$ is defined as

$$d_1 \approx_{\mathbb{R}^{n+1}, \delta} d_2 \iff ||b(d_1) - b(d_2)|| \leq \delta.$$

Similarly, $\approx_{E^\#, \delta} \in \mathcal{P}(\mathbb{D} \times \mathbb{D})$ is defined as

$$d_1 \approx_{E^\#, \delta} d_2 \iff e^\#(e \cdot \phi(d_1), e \cdot \phi(d_2)) \leq \delta.$$

then $\approx_{\mathbb{R}^{n+1}, \delta} \subseteq \approx_{E^\#, 2 \cdot \epsilon + K \cdot \delta}$ holds by above theorem. Thus it is quite reasonable to use $\approx_{\mathbb{R}^{n+1}, \delta}$ to cluster events, instead of unco[(b)]TJ/F283 8.966

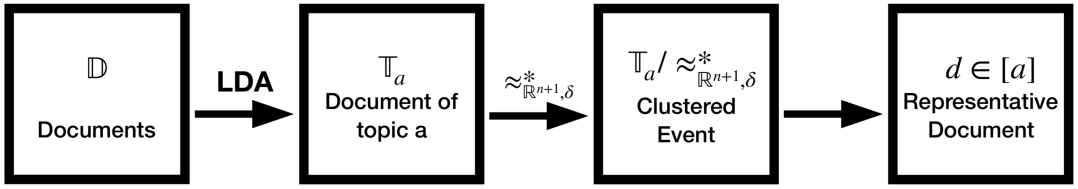

Figure 3: Overview of off-issue tracking process.

4.5 Extracting representative description

Now we have cluster of events (documents which describing events) $\mathbb{T}_a/\approx^*_{\mathbb{R}^{n+1},\delta}$, but we should return summary of events, because whole collection of documents are quite long to read and might have unnecessary information. So we have to extract *representative description* of tht event cluster. To extract target information from a document is well studied in information extraction field, and there are several method such as template-based information extraction, neural methods, etc. But in the case of several docuemnts, it is hard to converge summary to cover all document's information, because existing works is not based on language semantic-based, so it is hard to generate summary statement bewtween description of similar/same meaning.

For example, if one document describe the event happens "one day after of 12/7", and there are another document describe the event was happend "one day before of 12/9". Obviously both description refer same day, but token-based approach (or pattern-based approach such as signal words) cannot handle this issue. Even with this disadvantage, above method is widely used because of its high performance (and due to challenges of semantic based information extraction method).

So, we decided to use event extractor for one document, but we design to choose representative document appropriately.

Definition 4.8 (Representative docuemnt). document $d \in [a]$ is *representative document* of $[a]$ when $\sum_{d' \in [a]} ||b(d) - b(d')||$ $\leq \sum_{d' \in [a]} ||b(x) - b(d')||$ for any $x \in [a]$.

It means that we choose to extract event from a document which has minimum difference between all other documents. After choosing representative document, we use Giveme5W1H framework[**?**] to extract description of event.

4.6 implementation

To implement BoW transformation and document clustering, we use pandas and gensim for python. to calculate transitive closure and finding partition, we use DBSCAN algorithm. Parameters are adjusted by experiments on small set of documents. After that, extracting event description is done by Giveme5W1H framework.

5 EVALUATION

5.1 Trend analysis evaluation

To evaluate our result, we first try to show that our trend analysis works well. To do that, we collect other document with topic label. With these test set, its trend analysis result indirectly shows our accuracy of trend analysis.

5.2 Selecing test set

We use reuters data set. It consists of more than 9000 documents with more than 70 topics. But, the similairty of document is important because evaluation on very diffrent set of doucments dosen't imply any meaningful result. To resolve that, we decided to extract 10 topics with most similairty between our dataset. It is achived by calculating document similarity between reuters dataset and our news dataset.

To pick most similar topic, we compute maximum similairty within documents in topic and minimum similairty bewteen reuters and news dataset. Due to largeness of dataset, we choose only subset of dataset to calculate minimum/maximum similairty bewteen groups.

5.3 Reuters evaluation

With reuters dataset with 10 pre-classified topics, we generate LDA model for reuters dataset and make 10 topics. And we classified the topic with given label. In result, we successfuly classified 7 topics from LDA result. It shows that our LDA model seems to work correctly.

6 CONCLUSION

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

REFERENCES

[] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. 2017. NeuroNER: an easy-to-use program for named-entity recognition based on neural networks. *Conference on Empirical Methods on Natural Language Processing (EMNLP)* (2017).

[] Felix Hamborg, Corinna Breitinger, and Bela Gipp. 2019. Giveme5W1H: A Universal System for Extracting Main Events from News Articles. In *Proceedings of the 13th ACM Conference on Recommender Systems, 7th International Workshop on News Recommendation and Analytics (INRA 2019)*.

[] Katsiaryna Krasnashchok and Salim Jouili. 2018. Improving Topic Quality by Promoting Named Entities in Topic Modeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, 247–253. https://doi.org/10.18653/v1/P18-2040

[] Yingxu Wang. 2013. On Semantic Algebra: A Denotational Mathematics for Natural Language Comprehension and Cognitive Computing. *Journal of Advanced Mathematics and Applications* 2, 2 (2013), 145–161.

A APPENDIX

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.