Constructing knowledge graph from unstructured text

Kundan Kumar(12375) Siddhant Manocha(12714)

Under the guidance of Dr. Amitabh Mukherjee

Department Of Computer Science and Engineering
India Institute of Technology Kanpur

Abstract

A knowledge graph is a structured graphical representation of semantic knowledge and relations where nodes in the graph represent the entities and the edges represent the relation between them. Constructing a knowledge graph involve extracting relations from unstructured text followed by efficient storage in graphical databases.In this project, we propose a method for extracting relations using semantic regularity in the distributed word vector embedding space. Such a semi-supervised approach is independent of language syntax and can be used to extract relations from any language. We also investigate various similarity metrics to annotate each extracted relation with a confidence score. We use 'Neo4j' graphical database for efficient storage of extracted relations and constructing a knowledge graph. We further build a question answering system that parses the natural language queries using regular expressions and extracts answers from the knowledge graph.

Contents

1 Motivation

"It's time to move from Big Data to Big Knowledge"- Kevin Murphy (Google Research)[10]

The world wide web is a vast repository of knowledge, with data present in multiple modalities such as text, videos, images, structured tables, etc. However, most of the data is present in unstructured format and extracting information in structured machine-readable format is still a very difficult task. Knowledge graphs aim at constructing large repositories of structured knowledge which can be understood by machines. Such knowledge graphs are being used to improve the relevance and the quality of search in case of search engines like Google and Bing. Knowledge graphs are also being used by applications like Google now, Microsoft Cortana and Apple Siri which are capable of understanding natural language queries and answer questions, making recommendations, etc. to the user. The construction of knowledge graphs is thus a major step towards making intelligent personalized machines.

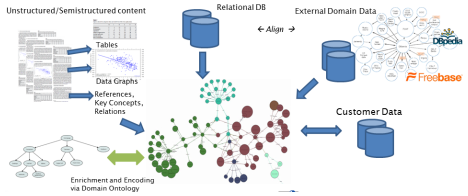

Figure 1: Extracting structured graph from unstructured data [1]

2 Previous Work

Last decade witnessed significant progress in the field of information retrieval and relation extraction, including projects like Never Ending Language Learning (NELL), OpenIE[4], YAGO, and Google Knowledge Vault. These projects proposed various methodologies to extract new structured information from the web, which has transformed the information on the web from 'strings' to 'things'[6].
Various methodologies for relation extraction can broadly be classified under the following three categories[6]:

2.1 Supervised approaches

Supervised models used in the field of information extraction involve formulation of the problem as a classification problem and they generally learn a discriminative classifier given a set of positive and negative examples. Such approaches extract a set of features from the sentence which generally include context words,

part of speech tags, dependency path between entity, edit distance, etc. and the corresponding labels are obtained from a large labelled training corpus.
Though such methods obtain good accuracy and takes into account for negative examples explicitly for relation extraction, they are neither general nor scalable. Such methods are very expensive due to requirement of large amount of training data. Moreover, the relations learned from these methods is largely dependent on the domain and thus cannot be generalized.

2.2 Semi supervised approaches

: Semi supervised approaches have been used for a long time in the field of relation extraction from unstructured text. Bootstrapping methods for relation extraction fall in this category. Such methods start with some known relation triples and then iterate through the text to extract patterns that match the seed triples. These patterns are used to extract more relations from the data set and the learned relations are then added to the seed examples. This method is repeated till no more relations can be learned from the data set. Some of the most popular approaches as Dual Iterative Pattern Relation Extractor[Brin et al][7], Snowball, Text Runner are examples of semi supervised methods. These projects, however, rely heavily on the correctness of NLP tools like Named Entity Recognition and thus they may be prone to errors. For example, TextRunner[2] extracts relations automatically from the text corpus using NLP tools like dependency parser and Noun Phrase chunker. Semi supervised approaches can be used to learn relations of a particular type accurately. However, seed examples are needed for that particular relation type and thus may require larger supervision for learning general knowledge graphs from different domains.

2.3 Distant supervision

: In case of distant supervision methods for relation extraction, existing knowledge bases are used with large text corpus to generate a large number of relation triples. Such relations are located in the text and hypotheses are learnt corresponding to these examples. These hypotheses can be combined to learn a generalized model for relation extraction.
Projects e.g. NELL use distant supervision methods for learning relations. They use predefined ontology and then bootstrap relations from web and text using positive and negative seed examples of ontology constraints. Later, they use multi-view learning paradigm to extract entities relations from unstructured text and web. Use of multiple hypotheses and an objective function based on agreement and disagreement of these hypotheses ensure less noise while expansion[8]. Often these hypotheses are hand-crafted and coming up with them require domain expertise.
These methods have distinct advantages over other methods since they are scalable and require almost no supervision. These models can also be used to learn relations from general domains. However, such methods are computationally

expensive and may learn wrong relation-expression hypotheses which can generate a large number of false relations.

Most of the above stated methods for relation extraction utilizes syntactic information either in the training phase or in the evaluation. Hence, these methods can only be used to learn relations from English corpus and fails for other languages like Hindi, for which there are no efficient and reliable syntactic NLP tools. As a result, we explore distributed word vector representations for learning relations which does not rely on syntactic information and extract semantic meaning of words based on their context. Thus, there is a scope to extend these methods to extract relations from corpus in any language.

3 Word Vector Representation

A word embedding is a functional mapping that maps the word in a particular language to some high dimensional space such that the semantic meaning of the words can be captured in this vector space. The state of the art algorithms for obtaining word vector embeddings like google word2vec and glove are computationally cheap and are capable of capturing the semantic meaning of the words in the word embedding space. One such approach for obtaining word vector representations namely continuous bag of words model is stated below.

3.1 Continuous Bag of Words Model

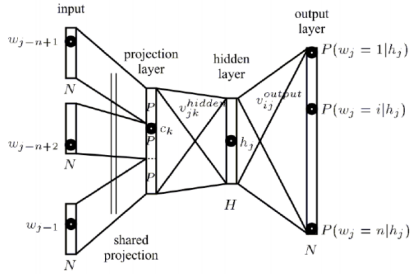

Figure 2: Continuos Bag Of Words Model[11]

The 'continuous bag of words model' (CBOW) has been proposed by Mikolov et al to learn distributed word vector embeddings. These architectures use the context of word in a sentence to predict the current word and learns the corresponding word representations. It is a simple architecture which does not contain any hidden layers. For the given architecture, let us assume that the vocabulary size is N and the word vector embedding size is V. Let the word

vector embeddings be represented by W , where W is a $N \times V$ matrix. The specifications of the architecture are:

1) Input: The left and the right context of the word are given as input in one-hot vector format i.e representation has a 1 corresponding to the context word and rest are all zeros i.e the each input vector $\{x_1, x_2 \dots x_N\}$,for the given input word only one entry is 1 and rest are 0.

2) Projection Layer:For the projection layer, instead of taking the input vectors directly, we take the average over the input context words and then use the weight vector embedding matrix to obtain input -¿ output

$$h = \frac{1}{C} W(x_1 + x_1 \dots x_C),$$

h is

where C is the number of context words.

3) The final layer is a softmax layer, which projects the hidden projection layer to $N \times 1$ softmax layer which models the probability with which each word occurs in the given input context. In the training the actual output word is known to us and we aim to maximize the probability of obtaining the given output word.

For learning the word representation, we generally use hierarchical softmax instead of softmax since it increases the computational efficiency.

Since these methods rely of the word context for mapping the word to vector space, it is quite intuitive that the words that appear in similar context will map to points that are close together in the word embedding space. Thus they are capable of capturing the semantic similarities between words in the language

3.2 Semantic Similarities

It has been observed in SemEval task 2012 that word vector representation (i.e. continuous vector space language model) performs surprisingly well in measuring semantic similarity of given relations. It was also shown that cosine similarity metric with vector offset method have been successful in solving analogy problems[3]. Mikolov et. al. (2013) stated in their paper that- "the learned word representations in fact capture meaningful syntactic and semantic regularities in a very simple way. Specifically, the regularities are observed as constant vector offsets between pairs of words sharing a particular relationship. For example, if we denote the vector for word i as xi, and focus on the singular/plural relation, we observe that $x_{apple} - x_{apples} \approx x_{car} - x_{cars}, x_{family} - x_{families} \approx x_{car} - x_{cars}$, and so on". If they are performing remarkably well for measuring semantic similarity amongst words and analogy task, the same can be exploited for the purpose of relationship extraction. Following image shows lower dimension representation of such relations:

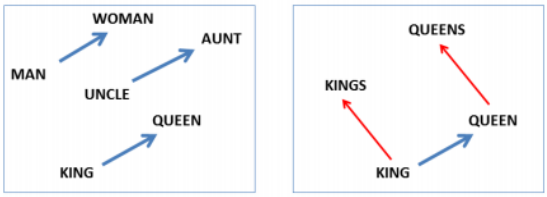

Figure 3: Left image shows vector offsets of words connected by gender relationship whereas right image shows singular-plural relationship for two words.[Image Source: Mikolov, Yih, Zweig (2013)]

4 Relation extraction

4.1 Approach

We adopt a semi supervised approach for the purpose of relation extraction. Our approach is motivated by the analogy tasks used in the evaluation for word vector embeddings by Mikolov et al. As stated above, the semantic regularity of the word embedding space can be used for extracting meaningful relations. We start with some seed examples say 'Spain' and 'Germany' and try to extract more entities that represent the same concept as these seed examples. In the case of above examples, we extract names of countries from the word embedding space. Since country names appear in similar context, it is expected that they lie close together in the word embedding space. Once, the countries are extracted, they can be used to learn analogy as 'Spain' is to 'Madrid' is same as 'Italy' is to 'Rome' to learn relations of type 'capital'. Similar approach can be used to extract relation in any language given their corresponding word embeddings independent of the syntactic structure of the sentence in that particular language.

4.2 Methodology

We propose the following approach for extracting relation from word embedding space using an initial set of seed examples:
Input: Distributed word vector representation learned using wikipedia text corpus and seed examples of the relation (say r) of the form (e_{s1_j}, e_{s2_j}) where $j \in 1..k$ where we define primary concept set as S1 which contains e_{s1_j} whereas S2 contains e_{s2_j} s.t. $j \in 1..k$. Output: Set of pairs (e_{g1_i}, e_{g2_i}) such that e_{g1_i} and e_{g2_i} have relation r between them. Following are the major steps of our algorithm:

1. Expand the primary concept to which e_{s1_j} belong: This is done by first considering the neighborhood of elements of S1 in the vector space $\forall j \in 1..k$ and taking their intersection. Say the intersection set is E. We calculate the score of each extracted entity in set E(Scoring criteria will be

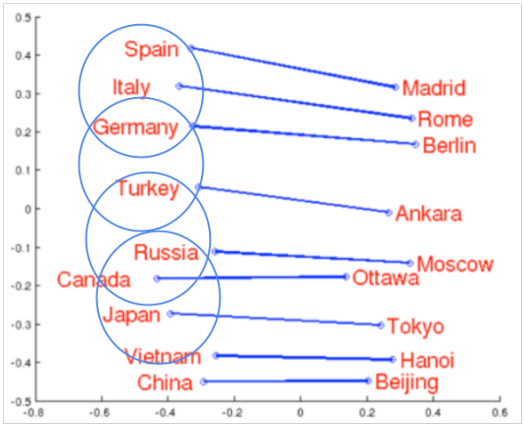

Figure 4: Approach:Relation Extraction

described in the following section). We take the entity with most confident (i.e. the entity with maximum score) and add to the set S1. Since we add one element at a time, noise is comparatively less and entities extracted tend to be correct. We continue this process until E is a null set.

2. Once step 1 terminates, we have sufficient number of entities of the concept representing the primary concept. Now let's say $e'_{g1_i} \in S1$ which was not present in seed examples. We get corresponding e'_{g2_i} by finding the nearest vector to $e'_{g1_i} - e_{s1_j} + e_{s2_j}$. We continue this process and if we get same e'_{g2_i} for different j i.e. for different seed relations then we add (e_{g1_i}, e_{g2_i}) to output. Now, we add output set to seed relations and continue till we do not get any new relation.

3. Output the output set obtained in step 2

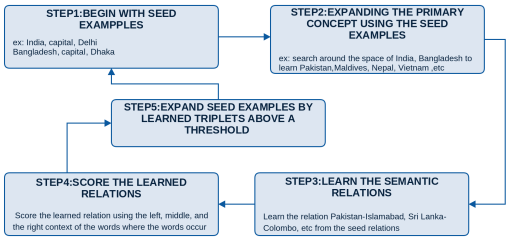

Figure 5: Algorithm flow chart

4.3 Scoring criterion

Following methods were used in order to score the extracted relations:

1. Context Based Methods: This method tries to find the similarity between two relation triples based on the context in which the linked entities occur in the corpus.

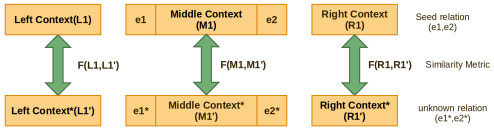

Figure 6: Context Based Methods for Similarity[5]

Since the relationship between the entities can be characterized by the context in which these relations occur, we try to extract the left, middle and the right context for the relations and try to find the similarity between these context. The similarity between the context is a good representative for finding the similarity between the relations. The context is represented by averaging out the word vectors that constitute the context and we calculate the cosine distance between the context to get an estimate of the similarity.

We used simple cosine similarity metric to find similarity between extracted concept entity and seed examples. For scoring, we tried taking average of the cosine similarity with all seed examples and also minimum of the cosine similarity with all seed examples.

Sentence1: 'his actions in Brecko' : his -> actions <- in <- Brecko
Sentence2: 'hi arrival in Beijing': his -> arrival <- in <- Beijing

his PRP Person	→	actions NNS Noun	←	in IN	←	Brecko NNP Noun Location	
his PRP Person	→	arrival NN Noun	←	in IN	←	Beijing NNP Noun Location	
3	1	1	1	2	1	3	18

Figure 7: Tree Kernel Based Approach for Similarity[9]

2. Syntactic approach(Tree Kernels): These methods extract the sentence where the given relation exist in the corpus and build a dependency parser[9] for that sentence. The path between the entities of the relations is then used for estimating the similarity between the relations. Since

we had a large corpus, it was not possible to parse all the sentences in which the extracted relations occur. Thus,we tried this approach on a smaller corpus. However using this metric defeats the purpose of a syntax independent approach to relation extraction.

3. We also tried using cosine similarity between relationship vector where relationship vector is defined by the vector joining the point representation of entities present in a relation in the word embedding space

5 Question Answering System

Question Answering systems are large scale systems that aim at finding answers to natural language queries from large amount of data present in the web. There is a clear distinction between traditional information retrieval systems and question answering systems. Information retrieval system aim at extracting relevant documents from the large collection which may contain a particular keyword. On the contrary, question answering systems aim at finding concise answers to natural language queries which may be present at a particular location in the document. As a result, question answering system requires greater analysis and processing of the user query for obtaining answers. In the given project, we utilize the constructed knowledge graph for obtaining answers to simple natural language queries. The question answering system consists of the following three components:

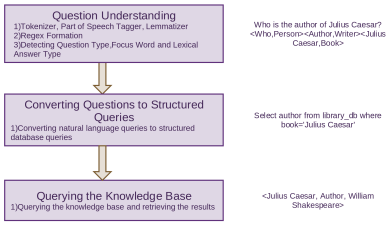

Figure 8: Question Answering System

- Question Understanding: This is the most important aspect of question answering systems. Question understanding involve detecting the type of question, understanding the focus or the target words in the question and the type of answer expected by the query. For this purpose, we pre process the query and then use regular expressions for extracting question words and target words from a given natural language query.
  The pre processing consists of the following steps
  - A tokenizer: It splits the sentence into tokens

- A part-of-speech tagger : It tags the part of speech for various tokens in the sentence
- A lemmatizer: It groups together different inflected form of the words so that they can represented as a single concept or lemma Then regular expressions are written over these POS tagged tokens and lemmas to obtain the question and the target word.

- Converting questions to structured queries: Once the natural language query in understood, this step involves using the extracted keywords to formulate structured queries that can be used to extract relevant answers. This step generally involves formulation of structured database queries which can then be used to extract relevant information from the database. In the given project,we convert the keywords into a structured neo4j database query.

- Answer extraction: This step involves querying the database to extract relevant answers and convert them into human readable format.
  The neo4j database query obtained in the previous part is used to query the knowledge graph and extract relevant answers.

6 Extracting Information from Wikipedia Infobox

Figure 9: Wikipedia Info box of Canada. [Image Source: Wikipedia.org]

Wikipedia has structured info-box which is a rich and reliable source of knowledge. It gives data about the edges associated with entities in a knowledge graph and also helps in discovering new edges and nodes for the knowledge graph. The information extracted from info box may be used as seed examples for many relations in our algorithm. A simple python script can be used

to extract information from these boxes by just giving the entity name e.g. “Canada”.

7 Results

Country	Language	Confidence
India	Hindi	1.0/S
France	French	1.0/S
Croatia	Croatian	0.81/P
Austria	German	0.75/P
Belgium	Dutch	0.78/P
Serbia	Polje	0.64/N
Poland	Polish	0.85/P
Moldova	Romanian	0.72/P
Slovakia	Czech	0.55/N
Belarus	Belarusian	0.57/P

Table 1: Confidence for extracted relation using seed examples. Confidence is represented by score/X where X=S implies it is a seed example, X=P, implies extracted relation is indeed true, X=N implies the extracted relation is false

Relation(Type)	Correct	Incorrect
Demonym	19	9
Language	25	10
Capital	21	11

Table 2: Performance: This table represents number of correct and incorrect relations extracted with two seed examples for three types of relations

8 Observations

Some of the key observations from the results and the experiments are as follows:

- We can obtain a threshold on confidence score to separate the positive relations from the negative ones. However, there exist a trade-off between the precision and recall, since increasing the threshold increases the precision but simultaneously reduces recall and vice-versa. Smaller threshold can learn large number of relations but also learns certain amount of false relations while larger threshold decreases the false relations but learns fewer relations. Hence an optimum value of theta that helps in expanding well while keeping accuracy significantly high is important.

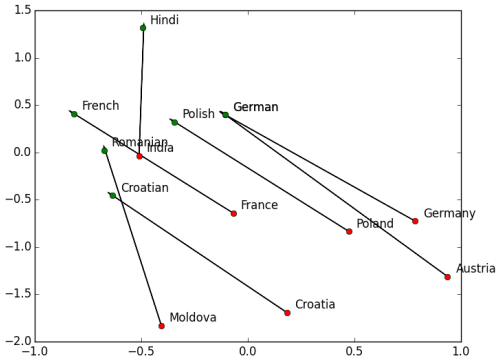

Figure 10: Lower dimensional embedding of country-language relationship

- The results have very high dependence on corpus used. Since we aimed at building knowledge graph, and wikipedia is assumed to be a good source of knowledge, we used wikipedia corpus for training word vector model. However, we also tested our model on Google News Corpus where performance was very poor in terms of extracting information on geographical entities.
- Scoring criteria used was average cosine similarity of vector offset of extracted relations with seed relations. Averaging out often doesn't work well. Taking maximum of cosine similarity with seed examples appear to be a good option but it has a problem that if even a single incorrect relation is added in seed examples (during the run of the algorithm), many incorrect examples (related to the already existing incorrect example) began populating the output. Hence, the algorithm fails to perform well. In some cases, taking maximum of cosine similarity with seed examples work very well while it fails miserably in many.
- The threshold on the confidence scores varies across different relation types and have to be set manually. This increases the level of supervision required in the model.

9 Discussions

- We have currently used only two seed examples. In the current version of our algorithm, more number of seed examples can easily be accommodated. With larger number of seed example, algorithm runs slow but it is expected to give better results.
- There was an interesting error saying "Jalpaiguri" is the capital of "Burma".

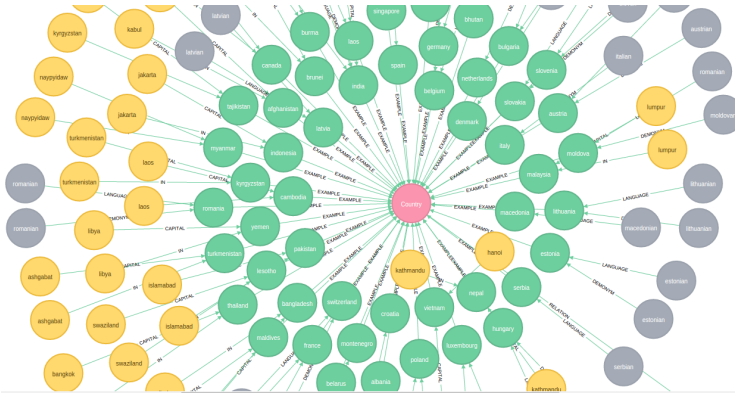

Figure 11: Snapshot of Knowledge Graph

Figure 12: Snapshot of Prototype Question Answering system

However, when we tried to see the capital of "Myanmar", it gave "Naypyidaw" which is true. It can be observed that Burma is often used in historical context for the country and hence has a strong relation with northeast India. This can be seen by observing the nearest neighbors of "Burma" in the vector space which include "Sikkim", "Bhutan", "Nepal","Assam","Myanmar", etc. Hence, it may be expected that "Jalpaiguri", being a north-eastern state may appear as a capital of "Burma".

10 Conclusion

It can be concluded that word vector representation, indeed, provides a great opportunity for relation extraction task. However, right choice of training cor-

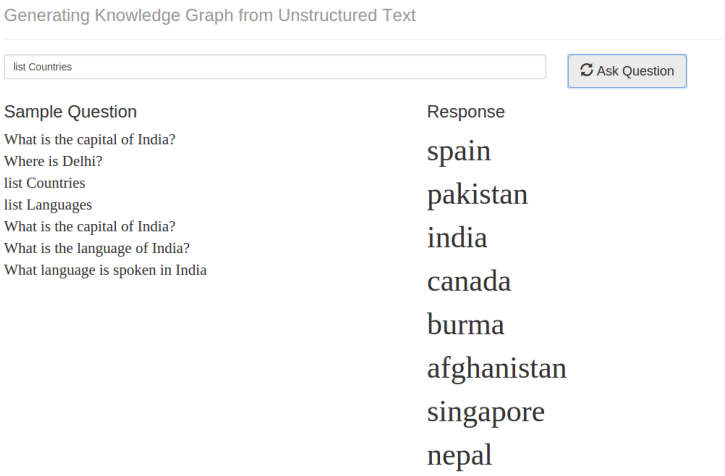

Figure 13: Snapshot of Prototype Question Answering system

pus is a must. Also, good scoring and evaluation metric for relations leads to better performance of the relation extraction algorithm. Expansion of concepts combined with relation extraction can be used to build a knowledge graph. This knowledge graph may be used to answer simple natural language queries. Regular grammar may be used to convert simple natural language questions to structured queries for the knowledge graph.

Acknowledgement

I take this opportunity to express my profound gratitude and deep regards to Dr.Amitabha Mukerjee for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him time to time shall carry me a long way in the journey of life on which I am about to embark. My thanks and appreciations also go to my friends who helped me during the course of the project and people who have willingly helped me out with their abilities. I am highly indebted to Tomas Mikolov for his publicly available Word2vec code.

Kundan Kumar,Siddhant Manocha
April,2015
Indian Institute of Technology,Kanpur

References

[1] Knowledge Graphs. http://www.sindicetech.com/overview.html.

[2] *TextRunner: Open Information Extraction on the Web*, Rochester, New York, USA, April 2007. Association for Computational Linguistics.

[3] *Linguistic Regularities in Continuous Space Word Representations.*, 2013.

[4] Association for Computational Linguistics. *Identifying relations for open information extraction*, 2011.

[5] N. Bach and S. Badaskar. A survey on relation extraction.

[6] A. Bordes and E. Gabrilovich. Constructing and mining web-scale knowledge graph,kdd 2014.

[7] S. Brin. Extracting patterns and relations from the world wide web. In *The World Wide Web and Databases*, pages 172–183. Springer, 1999.

[8] A. Carlson, J. Betteridge, E. R. Hruschka Junior, and T. M. Mitchell. Coupling semi-supervised learning of categories and relations. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing*, pages 1–9, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[9] A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics, 2004.

[10] G. R. Kevin Murphy. From big data to big knowledge.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.