Towards The Ontology-Based Approach For Factual Information Matching

Nataliia Sharonova
Department of Intelligent Computer Systems
National Technical University
"Kharkiv Polytechnic Institute"
Kharkiv, Ukraine
sharonova@kpi.kharkov.ua

Anastsiia Doroshenko
Department of Intelligent Computer Systems
National Technical University
"Kharkiv Polytechnic Institute"
Kharkiv, Ukraine
doroshenkoanastasiia@gmail.com

Olga Cherednichenko
Department of System Engineering and Management Information Systems
National Technical University
"Kharkiv Polytechnic Institute"
Kharkiv, Ukraine
olha.cherednichenko@gmail.com

***Abstract*— Factual information is information based on facts or relating to facts. The reliability of automatically extracted facts is the main problem of processing factual information. The fact retrieval system remains one of the most effective tools for identifying the information for decision-making. In this work, we explore how can natural language processing methods and problem domain ontology help to check contradictions and mismatches in facts automatically.**

Keywords*— *fact; natural language processing; information extraction; ontology; reference model; software

I. INTRODUCTION

Today, the fact retrieval system is one of the most effective tools for identifying information for decision-making. When you refer to something as a fact you mean that you think it is true or correct. Factual information is information based on facts or relating to facts. The reliability of automatically extracted facts is the main problem of processing factual information. It is especially important because of increasing density of text information flow in mass media and various social networks, forums and blogs [1, 2]. Different interpretations of the same phenomenon, as well as the inconsistency, inaccuracy or mismatch in information coming from different sources lead to the task of factual information extraction.

Despite the widespread use of multimedia, text remains one of the main types of information in most electronic stores [3, 4]. The development of effective approaches to the processing of texts for the purpose of filtering, forming a semantic portrait, navigating through the text database is one of the most topical areas of modern information technology. Factual information is a collection of factual facts, factual data, factual records, and so on. In turn, the concept of fact is understood as any information that, after removing them from the context, retains an independent meaning.

Facts should be distinguished from data that fixes the specifics of the object, the conditions of observation, etc. The concept of the scientific fact presupposes the elimination of such information, that is, it requires a certain generalization of the direct data. However, it is noted that there is no clear distinction between these. Only knowledge that has withstood a critical test, that is, obtained as a result of generalization and processing of data by abstract-logical thinking (of course, one must, be given the account that the attainment of absolutely reliable knowledge is only an ideal of the development of science, is practically unattainable).

Based on the fact definition [5, 6], it is possible to define the minimal semantic unit of factual search, which is a triad: agent-predicate-value. That is, the record of factual information must include a pointer to the fact search agent, the attribute or predicate of this object, and give a specific value of this attribute.

Such a definition makes it possible to extract concepts from weakly structured text sources of information and to represent relations between them in a structured way. The resulting structure is facts, both in the form of fairly simple concepts: keywords, personalities, organizations, geographical names, and in a more complex form, for example, the name of the person with her job and occupation.

II. STATE OF THE ART

There are number of approaches to information extraction from natural language texts [7, 8]. We can highlight lack of automated semantic understanding and low consistency of extracted facts.

Machine-learning models can be broadly classified as either generative or discriminative. Generative methods seek to create rich models of probability distributions and are so called because, with such models, one can 'generate' synthetic data. Discriminative methods are more utilitarian, directly estimating posterior probabilities based on observations. Compared to generative models, which can become intractable when many features are used, discriminative models typically allow the use of more features. Logistic regression and conditional random fields (CRFs) are examples of discriminative methods, while Naive Bayes classifiers and hidden Markov models (HMMs) are examples of generative methods.

Some common machine-learning methods used in natural language processing (NLP) tasks, and utilized by several articles in this issue [9, 10, 11], are summarized further.

Support vector machines (SVMs), a discriminative learning approach, classify inputs (eg, words) into categories (eg, parts of speech) based on a feature set. The input may be transformed mathematically using a 'kernel function' to allow *linear separation* of the data points from different categories. That is, in the simplest two-feature case, a straight line would separate them in an X–Y plot: in the general N-feature case, the separator will be an (N−1) hyperplane. The commonest kernel function used is a Gaussian (the basis of the 'normal distribution' in statistics). The separation process selects a *subset* of the training data (the 'support vectors'-data points closest to the hyperplane) that best differentiates the categories. The separating hyperplane maximizes the distance to support vectors from each category (as you can see on the slide).

An HMM is a system where a variable can switch (with varying probabilities) between several states, generating one of several possible output symbols with each switch (also with varying probabilities). The sets of possible states and unique symbols may be large, but finite and known. We can observe the outputs, but the system's internals (ie, state-switch probabilities and output probabilities) are 'hidden.' Main problems are:

1) Inference: given a particular sequence of output symbols, compute the probabilities of one or more candidate state-switch sequences.

2) Pattern matching: find the state-switch sequence most likely to have generated a particular output-symbol sequence.

3) Training: givem examples of output-symbol sequence data, compute the state-switch/output probabilities that fit this data best.

Naive Bayesian reasoning extended to sequences; therefore, HMMs use a generative model. To solve these problems, an HMM uses two simplifying assumptions (which are true of numerous real-life phenomena):

1) The probability of switching to a new state (or back to the same state) depends on the previous N states. In the simplest 'first-order' case (N=1), this probability is determined by the current state alone. (First-order HMMs are thus useful to model events whose likelihood depends on what happened last.)

2) The probability of generating a particular output in a particular state depends only on that state.

These assumptions allow the probability of a given state-switch sequence (and a corresponding observed-output sequence) to be computed by simple multiplication of the individual probabilities. Several algorithms exist to solve these problems. The highly efficient Viterbi algorithm finds applications in signal processing, for example, cell-phone technology.

Theoretically, HMMs could be extended to a multivariate scenario, but the training problem can now become intractable. In practice, multiple-variable applications of HMMs (eg, NER) use single, artificial variables that are uniquely determined composites of existing categorical variables: such approaches require much more training data.

HMMs are widely used for speech recognition, where a spoken word's waveform (the output sequence) is matched to the sequence of individual phonemes (the 'states') that most likely produced it. HMMs also address several bioinformatics problems, for example, multiple sequence alignment and gene prediction. Eddy provides a lucid bioinformatics-oriented introduction to HMMs, while Rabiner (speech recognition) provides a more detailed introduction.

Commercial HMM-based speech-to-text is now robust enough to have essentially killed off academic research efforts, with dictation systems for specialized areas - eg, radiology and pathology-providing structured data entry. Phrase recognition is paradoxically more reliable for polysyllabic medical terms than for ordinary English: few word sequences sound like 'angina pectoris,' while common English has numerous homophones (eg, two/too/to).

CRFs are a family of discriminative models first proposed by Lafferty et al. An accessible reference is Culotta ; Sutton and McCallum is more mathematical. The commonest (linear-chain) CRFs resemble HMMs in that the next state depends on the current state (hence the 'linear chain' of dependency).

CRFs generalize logistic regression to sequential data in the same way that HMMs generalize Naive Bayes. CRFs are used to predict the state variables ('Ys') based on the observed variables ('Xs'). For example, when applied to NER, the state variables are the categories of the named entities: we want to predict a sequence of named-entity categories within a passage. The observed variables might be the word itself, prefixes/suffixes, capitalization, embedded numbers, hyphenation, and so on. The linear-chain paradigm fits NER well: for example, if the previous entity is 'Salutation' (eg, 'Mr/Ms'), the succeeding entity must be a person.

The relationship between Naive Bayes, logistic regression, hidden Markov models (HMMs) and conditional random fields (CRFs). Logistic regression is the discriminative-model counterpart of Naive Bayes, which is a generative model. HMMs and CRFs extend Naive Bayes and logistic regression, respectively, to sequential data (adapted from Sutton and

McCallum). In the generative models, the arrows indicate the direction of dependency.

CRFs are better suited to sequential multivariate data than HMMs: the training problem, while requiring more example data than a univariate HMM, is still tractable.

An ‘N-gram’ is a sequence of N items-letters, words, or phonemes. We know that certain item pairs (or triplets, quadruplets, etc) are likely to occur much more frequently than others. For example, in English words, U always follows Q, and an initial T is never followed by K (though it may be in Ukrainian). In Portuguese, a Ç is always followed by a vowel (except E and I). Given sufficient data, we can compute frequency-distribution data for all N-grams occurring in that data. Because the permutations increase dramatically with N-for example, English has 26^2 possible letter pairs, 26^3 triplets, and so on-N is restricted to a modest number. Google has computed word N-gram data (N≤5) from its web data and from the Google Books project, and made it available freely.

N-grams are a kind of multi-order Markov model: the probability of a particular item at the N^{th} position depends on the previous N^{-1} items, and can be computed from data. Once computed, N-gram data can be used for several purposes:

3) Suggested auto-completion of words and phrases to the user during search, as seen in Google's own interface.

4) Spelling correction: a misspelled word in a phrase may be flagged and a correct spelling suggested based on the correctly spelled neighboring words, as Google does.

5) Speech recognition: homophones (‘two’ vs ‘too’) can be disambiguated probabilistically based on correctly recognized neighboring words.

6) Word disambiguation: if we build ‘word-meaning’ N-grams from an annotated corpus where homographs are tagged with their correct meanings, we can use the non-ambiguous neighboring words to guess the correct meaning of a homograph in a test document.

N-gram data are voluminous - Google's N-gram database requires 28 GB but this has become less of an issue as storage becomes cheap. Special data structures, called N-gram indexes, speed up search of such data. N-gram-based classifiers leverage raw training text without explicit linguistic/domain knowledge; while yielding good performance, they leave room for improvement, and are therefore complemented with other approaches.

Algebra of finite predicates is used as a mathematical tool for describing discrete, determinate and finite objects or processes of the real world [12, 13]. We use this math scheme to represent knowledge extracted from natural language texts: text information objects; the entity of the subject domain, grammatical and semantic characteristics of the text units.

According to the specific task we can form subsets from the elements of the universe of processing information. On Cartesian products of these subsets we can define relations between elements. We use predicate function to model these relations. The core point of the proposed approach is the predicate of recognition.

Despite of existing data extraction solutions the task of extracting facts still is not solved.

III. RESULTS

In general, the isolation of facts from poorly structured textual information includes the following stages [14]:

1) Entity Extraction - extraction of words or phrases that are important for describing the meaning of the text (lists of terms of the subject domain, personalities, organizations, geographical names, etc.);

2) Feature Association Extraction - research of connections between the extracted concepts;

3) Event and Fact Extraction - extraction of essences, recognition of facts and actions.

And since for the construction of the triad of factual information it is necessary to select entities represented in the texts under different names, then the stage of resolving the co-reference resolution of the syntactic analysis, to determine the synonyms, the entities of interest, acquires special significance.

At this stage, such pronouns as "he", "she", "they", "him", etc. should be associated with their antecedents, correlated them with the name of the essence of the given subject area.

The central task of obtaining factual information is the second stage of processing, which is the extraction of information about the relationships between entities.

At the same time, in order to extract a certain fact, it is necessary to define a certain template that reflects semantic (or conceptual) links in the sentence. To specify such semantic relations it is proposed to use the grammar of semantic cases. For this purpose it is necessary to develop a rigorous model that links the information contained in the definition of semantic roles with elements of the surface structure of sentences of natural language.

We suggest a model of facts extracting based on the method of comparator identification. This method represents the extraction process as a human intelligent activity since a human looking through a text can easily determine whether it corresponds to the template or not and catch attributes of a fact. It is based on the relation between the words and the location of these words in the text. The page estimation is based on a data source model. The presence of different combinations of words in different combinations of elements of the web page is estimated.

We can propose the reference model to factual information retrieval and analysis. The main concepts are facts that are some knowledge about real-world objects, web-pages which contain text, indicators for representing attributes, and values of those attributes. The appropriate models must formalize the factual data processing.

In recent years the development of ontologies (explicit formal specifications of the terms in the domain and relations among them) has been moving from the realm of Artificial-Intelligence laboratories to the desktops of domain experts [12]. Ontologies have become common on the World-

Wide Web. The ontologies on the Web range from large taxonomies categorizing Web sites to categorizations of products for sale and their features.

Ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them. Ontology together with a set of individual instances of classes constitutes a knowledge base.

Ontology is typically built in more-or-less the following manner [15, 16]:

1) acquire domain knowledge. Assemble appropriate information resources and expertise that will define, with consensus and consistency, the terms used formally to describe things in the domain of interest. These definitions must be collected so that they can be expressed in a common language selected for the ontology;

2) organize the ontology. Design the overall conceptual structure of the domain. This will likely involve identifying the domain's principal concrete concepts and their properties, identifying the relationships among the concepts, creating abstract concepts as organizing features, referencing or including supporting ontologies, distinguishing which concepts have instances, and applying other guidelines of your chosen methodology;

3) flesh out the ontology. Add concepts, relations, and individuals to the level of detail necessary to satisfy the purposes of the ontology;

4) check your work. Reconcile syntactic, logical, and semantic inconsistencies among the ontology elements. Consistency checking may also involve automatic classification that defines new concepts based on individual properties and class relationships;

5) commit the ontology. Incumbent on any ontology development effort is a final verification of the ontology by domain experts and the subsequent commitment of the ontology by publishing it within its intended deployment environment.

There are some engineering tools for developing ontologies, which have to be considered. It is possible to conclude, that Protégé [17] is the best tool for creating and supporting ontologies.

IV. CONCLUSION

Factual analysis of the text is designed to make possible the intellectual analysis of data extracted from the text flow. The solution of this task should lead to a synergistic effect, to the possibility of using existing information technologies.

To sum up, we can say that factual analysis is a rather complex system that has great potential and functionality. The tasks are designed to facilitate the work of analysts, to carry out filtration as well as structuring of huge volumes of information, which in our time are one of the main tasks of a person.

As a result, we can underline that the task of identifying instances, relations, events and their relevant properties in natural language texts is still a pressing issue. In general, we consider two kinds of facts. Despite existing data extraction solutions the task of extracting facts has not been solved yet. We propose to use predicate algebra and method of comparator identification to create a model of searching and extracting factual data.

REFERENCES

[1] Álvaro Figueira, Luciana Oliveira, The current state of fake news: challenges and opportunities / Procedia Computer Science 121, 2017, pp. 817–825.

[2] Giovanni Luca Ciampaglia, Fighting fake news: a role for computational social science in the fight against digital misinformation / J Comput Soc Sc (2018) 1:147–153, https://doi.org/10.1007/s42001-017-0005-6

[3] E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret and L. de Alfaro, "Some Like it Hoax: Automated Fake News Detection in Social Networks," Cornell University, New York, USA, 2017.

[4] B. Riedel, I. Augenstein, G. P. Spithourakis and S. Riedel, "A simple but tough-to-beat baseline for the Fake News Challenge stance detection task," Cornell University, New York, USA, 2017.

[5] Mauridhi Hery Purnomo et al., Biomedical Engineering Research in the Social Network Analysis Era: Stance Classification for Analysis of Hoax Medical News in Social Media / Procedia Computer Science 116, 2017, pp. 3–9.

[6] N. Rakholia and S. Bhargava, "“Is it true?” – Deep Learning for Stance Detection in News," Stanford University, California, USA, 2016.

[7] Khairova, N.F., Petrasova, S., Gautam, A.P.S.: The logical-linguistic model of fact extraction from English texts. In: Dregvaite, G., Damasevicius, R. (eds.) ICIST 2016. CCIS, vol. 639, pp. 625–635. Springer, Cham (2016).

[8] Khairova N., Lewoniewski W., Węcel K. (2017) Estimating the Quality of Articles in Russian Wikipedia Using the Logical-Linguistic Model of Fact Extraction. In: Abramowicz W. (eds) Business Information Systems. BIS 2017. Lecture Notes in Business Information Processing, vol 288. Springer, Cham

[9] Georg Rehm, An Infrastructure for Empowering Internet Users to Handle Fake News and Other Online Media Phenomena / An Infrastructure for Empowering Internet Users, 2017, pp. 216-231.

[10] N. J. Conroy, V. L. Rubin and Y. Chen, "Automatic Deception Detection: Methods for Finding Fake News," in Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community , Missouri, USA, 2015.

[11] Rubin, V., Conroy, N., and Chen, Y., Towards News Verification: Deception Detection Methods for News Discourse. 2015.

[12] Bondarenko M. F., Shabanov-Kushnarenko U. P. Theory of intelligence: a Handbook //SMIT Company, Kharkiv. – 2006.

[13] Nina Khairova, Natalia Sharonova. Use of Predicate Categories for Modelling of Operation of the Semantic Analyzer of the Linguistic Processor./Proceedinga of IEEE EAST-West Design & Test Symposium EWDTS'09 (2009).

[14] Sharonova N. et al. Issues of Fact-based Information Analysis / N. Sharonova, A. Doroshenko, O. Cherednichenko / Proc. 2nd Int. Conf. on Computational Linguistics and Intelligent Systems (COLINS), Volume I: Main Conference (Lviv, Ukraine, June 25-27, 2018). – CEUR-WS. – 2018. – Vol. 2136. – P. 11-19.

[15] Noy Natalya F. Ontology Development 101: A Guide to Creating Your First Ontology / Natalya F. Noy, Deborah L. McGuinness. — Stanford, California, Stanford University, 2001 – 25pp.

[16] Shvaiko Pavel. Ontology Matching / Pavel Shvaiko, Jerome Euzenat. — Berlin-Heidelberg, Springer-Verlag, 2007 – 333pp.

[17] The Protégé Ontology Editor and Knowledge Acquisition System // http://protege.stanford.edu