KAJIAN TEKNIK *KNOWLEDGE EXTRACTION* PADA BERBAGAI *TOOLS*

Firmansyah
Balai Pendidikan dan Pelatihan
Tambang Bawah Tanah
Sawahlunto

Yohanes Gultom
Faculty Of Computer Science
University Of Indonesia
Depok, Indonesia
Yohanes.gultom@ui.ac.id

1. PENDAHULUAN

Pada tahun 2012, IBM, salah satu perusahaan teknologi informasi terbesar di dunia, berhasil membuat terobosan dengan mendemonstrasikan kemampuan *Question Answering (QA) System* mereka untuk mengalahkan *human experts* pada kontes kuis *Jeopardy!*. Sistem QA yang diberi nama IBM Watson [1] tersebut menjadi salah satu indikasi bahwa perkembangan teknologi saat ini telah memungkinkan kita untuk membangun suatu sistem yang mampu otomatis mengekstrak pengetahuan umum dari dokumen lintas domain, membangun basis pengetahuan, menganalisis pertanyaan terbuka (*open question*) dan memilih jawaban terbaik dari basis pengetahuan yang ada.

Salah satu bagian penting dalam *QA System* secara umum dan IBM Watson secara khusus adalah reperesentasi formal pengetahuan atau basis pengetahuan (*knowledge base*) [2]. Untuk membangun sebuah basis pengetahuan, perlu dilakukan proses ekstraksi pengetahuan (*knowledge extraction*) dari sumber data yang dapat berupa teks (dokumen, buku, artikel), suara (rekaman pidato, kuliah, berita, percakapan), gambar (foto, diagram, grafik, peta) atau gabungan dari beberapa jenis data (video, ensiklopedia bergambar). Contoh dari *tools* untuk melakukan *knowledge extraction* adalah PRISMATIC [10], yang merupakan bagian dari IBM Watson.

Knowledge extraction adalah proses ekstraksi informasi beserta relasinya, melakukan generalisasi terhadap informasi tersebut serta menyimpannya secara terstruktur dalam *knowledge base* sehingga dapat dengan mudah diakses dan diinferensi [2] (contoh: mengekstraksi pengetahuan tentang tanaman pangan dari buku-buk teks). Sekalipun memiliki tujuan yang berbeda, *knowledge extraction* dapat menggunakan teknik-teknik *information extraction* yang bertujuan untuk mengekstraksi informasi (eksplisit) dengan kategori tertentu dari sekumpulan dokumen [5]

[2]http://hadoop.apache.org/

[3]http://lucene.apache.org

(contoh: mengekstraksi informasi harga pangan dari artikel-artikel berita). *Knowledge base* yang dihasilkan, nantinya dapat digunakan sebagai sumber daya untuk *task* lain seperti *knowledge discovery,* yang berfokus untuk mengekstrak informasi *nontrivial*, implisit, tidak diketahui sebelumnya dan potensial untuk digunakan dari data yang ada [4],

Pada kajian ini Penulis menelaah dan membandingkan berbagai penelitian terkait mengenai *tools knowledge extraction*, khususnya mengenai algoritma yang digunakan yang umumnya berupa *pipline* dari teknik-teknik *Natural Language Processing* (NLP). Tujuan dari kajian ini adalah untuk mengetahui kondisi terkini penelitian mengenai bidang *knowledge extraction* serta menganalisa karakteristik dan kinerja masing-masing *NLP-pipeline* digunakan pada masing-masing *tools*.

2. *TOOLS*

Pada bagian ini Penulis mengkaji lima buah *tools knowledge extraction*, terurut berdasarkan waktu publikasinya, yaitu DIRT, TextRunner, PRISMATIC, REFRACTIVE dan K-Extractor. Kajian ini akan memberikan sedikit gambaran menyeluruh (*overview*) tentang masing-masing sistem dan berfokus pada rangkaian *task* NLP (*NLP-pipeline*) yang digunakan.

2.1 DIRT

DIRT (*Discovery of Inference Rules from Text*)[6] adalah sebuah algoritma untuk mengekstrak aturan inferensi dari teks secara otomatis. Aturan inferensi yang dimaksud contohnya seperti ”*X is author of Y* ≈ *X wrote Y*”, ”*X solved Y* ≈ *X found a solution to Y*” dan ”*X caused Y* ≈ *Y is triggered by X*”. Algoritma ini sudah diimplementasikan dan diuji oleh penemunya sehingga dapat dikaji juga sebagai salah satu *tools*.

Algoritma DIRT menentukan ekstraksi aturan berdasarkan *dependency tree* yang dihasilkan oleh *statistical parser*, Minipar[8], dan menerapkan *extended distributional hypothesis*[6] untuk mengekstrak *path* dari *tree*. Jika *distributional hypothesis*[7] mengatakan bahwa kata-kata yang muncul pada konteks yang sama umumnya memiliki makna yang sama, maka versi *extended* ini mengasumsikan bahwa semua *path* pada *dependency tree* yang menghubungkan himpunan kalimat yang sama kemungkinan memiliki makna yang sama juga. Sebagai contoh:

- Kata ”*duty*” dapat dimodifikasi oleh kata-kata sifat seperti ”*additional*”, ”*administrative*”, ”*assigned*”, ”*assumed*”, ”*collective*”, ”*congressional*” dan ”*constitutional*”. Kata ”*responsibility*” juga dapat dimodifikasi oleh kata-kata sifat tersebut.

- Kata ”*duty*” bisa menjadi objek dari kata-kata kerja seperti ”*accept*”, ”*articulate*”, ”*assert*”, ”*assign*”, ”*assume*”, ”*attend to*”, ”*avoid*”, ”*become*” dan ”*breach*”. Demikian juga ”*responsibility*” dapat menjadi objek dari kata-kata tersebut.

- Berdasarkan *extended distributional hypothesis*, ”*duty*” dan ”*responsibility*” memiliki makna yang sama

Ekstraksi *path* dari *dependency tree* untuk menghasilkan *path* seperti $N : sub\,j : V \leftarrow buy \rightarrow : from : N$ (*X buy something from Y*) dilakukan dengan pendakatan *rules-based* dengan aturan sbb[6]:

- *Slot* X dan Y harus berupa *noun*

- *Path* harus menghubungkan dua konten (X dan Y)

- Frekuensi (jumlah kemunculan) dari *internal relation* harus melebihi *threshold*

Penentuan kemiripan *path* dilakukan dengan menghitung *mutual information* dari *slot* X dan Y dari *triples table* yang merupakan penyimpanan *path* yang telah diekstrak dan frekuensinya.

Hasil pengujian[6] terhadap DIRT menunjukkan bahwa dari korpus teks dari surat kabar (*AP Newswire, San Jose Mercury* dan *Wall Street Journal*) sebesar 1GB dapat dihasilkan 7 juta *paths* yang disimpan dalam *triple database*. Seluruh *Path* tersebut dihasilkan dari sekitar 231.000 *parse tree*.

Sekalipun tidak dinyatakan secara eksplisit dalam publikasinya, koleksi *path* yang diekstrak DIRT (*triples table*) sudah dapat disimpan sebagai *knowledge base* karena dapat sudah digunakan untuk melakukan inferensi. Sebagai salah satu algoritma yang melakukan *knowledge extraction* otomatis, DIRT juga menjadi acuan oleh sistem-sistem setelahnya seperti PRISMATIC[5].

2.2 TextRunner

TextRunner[9] adalah kakas *Open Information Extraction* (OIE) yang diajukan untuk menangani kasus *information extraction* yang lebih kompleks di mana kategori informasi yang diekstrak menjadi lebih terbuka (*open*) dan korpus yang ditangani bersifat heterogen (untuk menangani sumber dari *web*). Hasil ekstraksi dari kakas ini adalah koleksi *queryable tuples* $t = (e_i, r_j, e_k)$ di mana e_i,e_k adalah entitas dan r_j adalah relasi antara entitas tersebut.

TextRunner terdiri dari empat buah komponen penting[9]:

[2]http://hadoop.apache.org/

[3]http://lucene.apache.org

1. *Single Pass Extractor*

   Proses utama yang dilakukan oleh TextRunner adalah melakukan satu iterasi (*pass*) terhadap seluruh dokumen untuk melakukan *POS tagging* dan *noun-phrase chunking*. Jika ditemukan subjek dan dua buah *noun-phrase*, sebuah *classifier* akan digunakan untuk menentukan apakah *tuple* bisa dibentuk dari subjek dan *noun-phrase* yang ditemukan.

2. *Self-Supervised Classifier*

   Classifier yang digunakan untuk menentukan ekstraksi *tuple* ini merupakan model *machine learning* yang dilatih menggunakan data yang dihasilkan oleh *full-parser* berdasarkan fitur-fitur heuristik di antaranya *POS tag*.

3. *Synonym Resolver*

   Karena TextRunner tidak menggunakan daftar relasi ketika mengekstrak *tuple*, kemungkinan besar akan dihasilkan banyak *tuple* yang memiliki relasi yang bermakna sama. Merupakan tugas dari *synonym resolution* untuk melakukan *clustering* terhadap seluruh *tuple* yang dihasilkan sehingga dapat dikenali hubungan sinonimnya.

4. *Query Interface*

   TextRunner juga membangun *inverted index* dari seluruh *tuple* yang dihasilkan sehingga pencarian (*query*) dapat dilakukan dengan cepat berdasarkan entitas atau relasi.

Berdasarkan hasil pengujian, TextRunner mampu mengekstrak 7,8 juta *tuple* dari korpus yang berisi 9 juta dokumen. Sampel yang berisi 400 *tuple* yang dipilih secara acak dinilai 80.4% benar oleh ahli.

Karakteristik permasalahan dan teknik yang digunakan oleh TextRunner, membuat kakas ini juga dapat dikategorikan sebagai kakas *knowledge extraction*. Alasannya adalah koleksi *tuple* berisi entitas dan relasi yang sudah dikelompokkan berdasarkan sinonimnya ini menyerupai format *Resource Description Framework* (RDF) yang umum digunakan pada *knowledge base*.

Saat ini TextRunner sudah melalui beberapa tahap pengembangan dan dikenal dengan nama Open IE[1]. TextRunner saat ini disebut sebagai versi 1 dari Open IE yang sudah mencapai versi 4 dan distribusikan secara *opensource* dengan lisensi dari *University of Washington*.

2.3 PRISMATIC

PRISMATIC[2] adalah bagian dari sistem QA IBM Watson yang bertugas

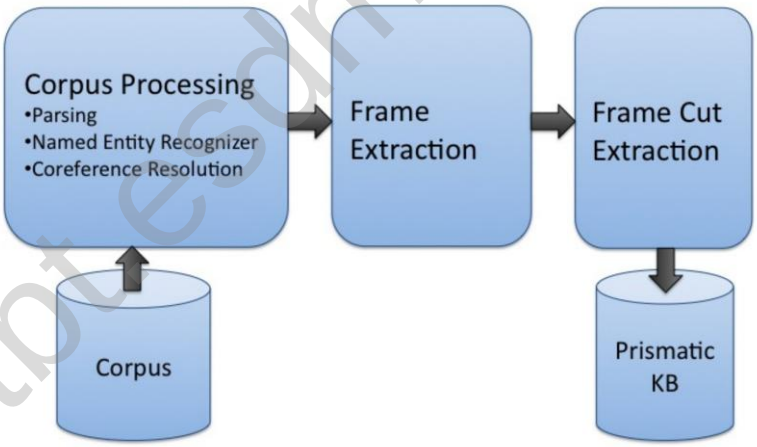
melakukan *knowledge extraction* yang berbasiskan *frame-slot*. Proses ekstraksi yang dilakukan oleh PRISMATIC, seperti yang ditunjukkan pada gambar 2.1, terdiri dari tiga tahap:

[1]http://openie.allenai.org/

1. *Corpus processing*
   Masukan dari tahap ini adalah dokumen dan hasilnya adalah dokumen yang sudah dianotasi menggunakan *dependency parsing* (tanpa *POS tagging*), *co-reference resolution*, *named-entity recognition* dan *relation detection.*

2. *Frame extraction*
   Mengekstraksi *frame* berdasarkan anotasi *dependency parsing* dan anotasi-anotasi lain yang berasosiasi dengannya.

3. *Frame projection*
   Melakukan proyeksi *frame* (*Subject-Verb-Object*) tertentu berdasarkan relasi untuk membentuk basis pengetahuan.



Gambar 2.1: Arsitektur PRISMATIC

Kunci utama pada tahap *corpus processing* adalah penerapan *dependency parser* menggunakan *English Slot Grammar* (ESG) *parser*[11]. Mirip seperti proses pada DIRT[6], *dependency parser* digunakan untuk mengisi *slot* hanya saja struktur yang digunakan bukan *path* tetapi *frame-slot*. Setelah itu untuk memastikan tidak ada informasi yang hilang, digunakan juga *rule-based co-reference resolution* sederhana. Kemudian *named-entity recognizer* (NER) digunakan untuk mengenali jenis entitas pada *frame-slot*.

Pada tahap *frame extraction*, hasil anotasi dari *corpus processing* kemudian disaring dan dimasukkan ke dalam *frame-slot* berdasarkan daftar relasi yang sudah didefinisikan (contoh: *subj*, *obj*, *pred*, *isa* .dsb).

[2]http://hadoop.apache.org/

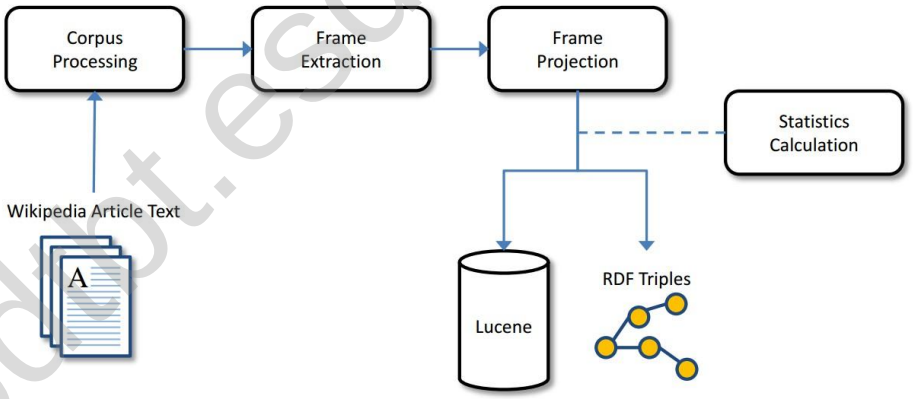
[3]http://lucene.apache.org

Pada tahap terakhir, dilakukan pemotongan *frame-slot* pada tahap *frame cut*, yaitu memilih kombinasi relasi tertentu seperti S-V-O, S-V-O-IO, S-V-P-O dsb (di mana S - subject, V - verb, O - object, IO - indirect object). Kombinasi yang telah dipilih itulah yang akan dimasukkan ke dalam *knowledge base* PRISMATIC.

2.4 REFRACTIVE

REFRACTIVE[12] adalah kakas *knowledge extraction opensource* yang terinsipirasi dari PRISMATIC. REFRACTIVE dirancang untuk mampu mengekstraksi *knowledge* dalam skala besar dengan mengandalkan teknologi *MapReduce* dari Apache Hadoop[2].

Seperti halnya PRISMATIC, proses *knowledge extraction* pada REFRACTIVE juga terdiri dari tiga tahap, yaitu *corpus processing*, *frame extraction* dan *frame projection*. Perbedaan utama pada REFRACTIVE adalah pada *task* NLP yang digunakan pada tahap *corpus processing*, yaitu *syntactic/dependency parsing*, *semantic parsing*, *named entity recognition* dan *co-reference resolution* (menggunakan *Multi-Pass Sieve Coreference Resolution*[15]). Gambar 2.1 menunjukkan arsitektur keseluruhan REFRACTIVE.



Gambar 2.2: Arsitektur REFRACTIVE

Syntactic parsing yang dilakukan pada *corpus processing* adalah (*dependency parsing*) dengan algoritma *passive-aggresive perceptron* yang diimplementasikan sebagai *Hash Kernel*[13]. Algoritma ini juga didesain untuk dapat diparalelisasi untuk meningkatkan kecepatannya.

Sedangkan *semantic parsing* yang dimaksud merupakan *pipeline* dari *L2-regularized linear logistic regression classifiers*[14] yang terdiri dari:

1. *Predicate identification classifier*

   Menentukan apakah *noun* atau *verb* merupakan predikat

2. *Predicate disambiguation classifier*
   Disambiguasi predikat yang memiliki banyak *sense*
3. *Argument identification classifier*
   Menentukan apakah suatu kata merupakan argumen
4. *Argument classification classifier*
   Menentukan jenis/kelas argumen (subjek atau objek)

Frame extraction dilakukan dengan memproyeksikan *sub-trees* dengan tinggi tertentu (*fixed height*) dari *dependency parse tree*. *Frame* yang diekstrak hanya yang *root*-nya berupa *noun* atau *verb*.

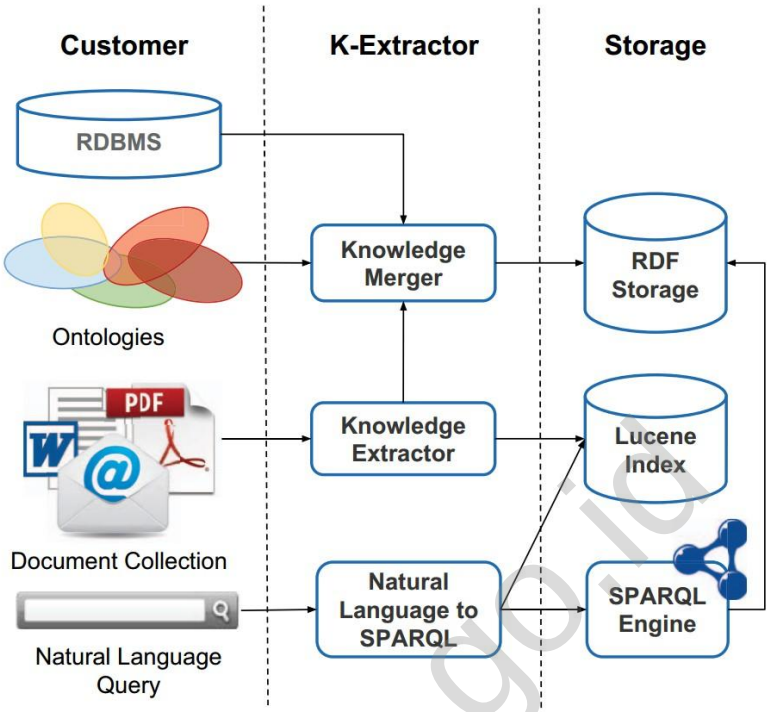
Pada tahap terakhir, *frame projection*, *frame* yang sudah dihasilkan disaring berdasarkan jenis slot nya (contoh SBJ, VERB, OBJ), dihitung statistiknya dan disimpan dalam *knowledge base*. REFRACTIVE juga menyediakan fitur untuk mengekspor *knowledge base* yang dibangun sebagai Lucene *index* atau RDF *triples*. REFRACTIVE mampu memproses 4 juta artikel dari Wikipedia (bahasa Inggris) dan berhasil mengekstraksi 260 juta *frame*. Waktu yang dibutuhkan untuk menyelesaikan proses tersebut menggunakan *cluster* 60 *cores* adalah 463 jam.

2.5 K-Extractor

K-Extractor[16] adalah sistem yang berfungsi untuk melakukan *knowledge extraction* dari koleksi dokumen, menggabungkan *knowledge* dengan data terstruktur dan menyediakan antarmuka *question answering* (QA) untuk mengakses *knowledge base* yang telah dibangun. Arsitektur K-Extractor secara keseluruhan dapat dilihat pada gambar 2.3.

[2]http://hadoop.apache.org/

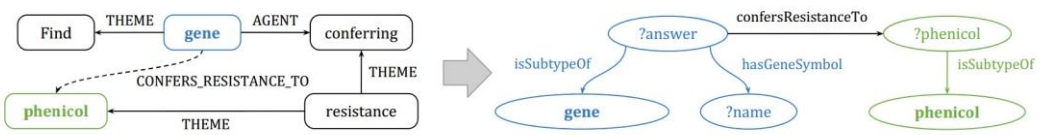
[3]http://lucene.apache.org



Gambar 2.3: Arsitektur K-Extractor

Komponen *knowledge extractor* pada K-Extractor bertugas untuk mengkonversi koleksi dokumen yang ada menjadi *semantic triples*[16]. Konversi dilakukan dengan memasukkan dokumen ke dalam *NLP-pipeline* yang terdiri dari *tokenization, POS-tagging, word sense disambiguation, named-entity recognition* (berdasarkan pola leksikal dan ontologi), ekstraksi *semantic relations, co-reference chains and events*.

Salah satu perbedaan proses ekstraksi *semantic relations* dibanding *tools* lainnya adalah ekstraksi dilakukan dalam dua tahap, yaitu ekstraksi umum (tanpa tema khusus) dan ekstraksi berdasarkan relasi yang menarik bagi pengguna[16]. Pada tahap pertama, ekstraksi dilakukan memberikan label yang mendefinisikan hubungan setiap kata. Kemudian pada tahap kedua, hasil dari tahap pertama disaring untuk mengambil *semantic relations* yang sesuai dengan ketertarikan pengguna (*domain specific*).



Gambar 2.4: Dua tahap *semantic extraction* pada K-Extractor

Hasil akhir *knowledge extractor* K-Extractor adalah *semantic triples* yang disimpan dalam *R*DF *storage*. Selain itu, RDF *semantic index* yang mencakup semua

konsep, *named-entity*, *co-reference chains*, *semantic relations* dan *event structures* digunakan juga sebagai suplemen pencarian *free-text index based* dengan Lucene[3].

2.6 Ringkasan

Berdasarkan kajian di atas, penggunaan NLP *task* pada kelima *tools knowledge extraction* dirangkum pada tabel 2.1.

Tabel 2.1: NLP *task* pada *tools knowledge extraction*

NLP *task*	DIRT	TextRunner	PRISMATIC	REFRACTIVE	K-Extractor
POS *tagging*		./		./	./
Chunking		./	./		
Synonym resolver		./			
Dependency parsing	./		./	./	
NER			./	./	./
Co-reference resolution		./	./	./	./
Relation detection			./		./
WSD				./	./

3.KESIMPULAN

Berdasarkan kajian yang telah dilakukan, Penulis menarik beberapa kesimpulan:

- NLP *task* yang berperan paling umum dalam *knowledge extraction* adalah *co-reference resolution* karena digunakan di semua *tools* kecuali DIRT. Sedangkan yang paling sedikit digunakan adalah *synonym resolver* yang mengindikasikan bahwa informasi sinonim tidak signifikan dalam *knowledge extraction*

- Representasi *knowledge* menggunakan *Resource Data Framework* (RDF) *triples* dinilai paling efektif dan efisien karena sederhana dan mudah diinferensi

- Terkait dengan dipilihnya RDF *triples* sebagai representasi *knowledge*, *task* NLP yang paling penting adalah deteksi *noun-phrase* dengan *named-entity recognition* yang diikuti *dependency parsing* atau *chunking*

- Pendekatan *machine learning* merupakan pendekatan yang dipilih oleh semua *tools* pada kajian ini karena dapat melakukan klasifikasi dan *clustering*

[2]http://hadoop.apache.org/

[3]http://lucene.apache.org

terhadap korpus data yang besar dengan lebih cepat dibanding pendekatan *rule-based*

- Paralelisasi *knowledge extraction* menggunakan algoritma seperti MapReduce pada *cluster* dapat dijadikan solusi untuk meningkatkan kecepatan dan kapasitas *knowledge extraction*

DAFTAR REFERENSI

[1] Ferrucci, D. et. al. *Building Watson: An Overview of the DeepQA Project*. 2010. Association for the Advancement of Artificial Intelligence.

[2] Fan, J, et al. *Automatic knowledge extraction from documents*. 2012. IBM Journal of Research and Development 56.3.4: 5-1.

[3] McCord, M. C., Murdock J. W. & Boguraev, B. K. *Deep parsing in Watson*, 2012. IBM J. Res. & Dev., vol. 56, no. 3/4, Paper 3, pp. 3:13:15.

[4] Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. *Knowledge discovery in databases: An overview*. 1992. AI magazine, 13(3), 57.

[5] Freitag, D. *Machine learning for information extraction in informal domains*. 2000. Machine learning, 39(2-3), 169-202.

[6] Lin, D., & Pantel, P. *DIRT - discovery of inference rules from text*. 2001. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 323-328). ACM.

[7] Harris, Z. *Distributional Structure*. 1985. In: Katz, J. J. (ed.) The Philosophy of Linguistics. New York: Oxford University Press. pp. 26-47.

[8] Lin, D. *MINIPAR: a minimalist parser*. 1999. In Maryland linguistics colloquium.

[9] Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., & Soderland, S. *Textrunner: open information extraction on the web*. 2007. In Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations (pp. 25-26)

[10] Fan, J, et al. *Prismatic: Inducing knowledge from a large scale lexicalized relation resource*. 2010. Proceedings of the NAACL HLT 2010 first international workshop on formalisms and methodology for learning by reading. Association for Computational Linguistics.

[11] McCord, M. C. *Slot grammar: A system for simpler construction of practical natural language grammars*. 1990. In Proceedings of the International Symposium on Natural Language and Logic, pages 118145, London, UK. Springer-Verlag.

[12] Exner, P., & Nugues, P. *REFRACTIVE: An Open Source Tool to Extract Knowledge from Syntactic and Semantic Relations*. 2014. In LREC, The 9th edition of the Language Resources and Evaluation Conference (pp. 2584-2589).

[13] Bohnet, B. *Very High Accuracy and Fast Dependency Parsing is not a Contradiction*. 2010. In Huang, C.-R. and Jurafsky, D., editors, COLING, pages 8997. Tsinghua University Press.

[14] Bjrkelund, A., Bohnet, B., Hafdell, L., and Nugues, P.A high-performance syntactic and semantic dependency parser. 2010. In Coling 2010: Demonstration Volume, pages 3336

[15] Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., and Manning, C. *A multi-pass sieve for coreference resolution*. 2010 In Proc. of EMNLP-2010, pages 492501, Boston.

[16] Balakrishna, M., Werner, S., Tatu, M., Erekhinskaya, T., & Moldovan, D. *K-Extractor: Automatic Knowledge Extraction for Hybrid Question Answering*. 2016. In 2016 IEEE Tenth International Conference on Semantic Computing (ICSC) (pp. 390-391). IEEE.