Yapılacaklar:

1. Templatelere göre textlerden veri çekilecek.
2. Template verileri ile infobox verilere karşılaştırılacak.
   1. Sorunlar
      1. Metin içinde geçmiyor bazıları?
         1. (http://tr.wikipedia.org/wiki/Bruce\_Cohen)
         2. (<http://tr.wikipedia.org/wiki/Carlos_S%C3%A1nchez_Moreno>)
      2. Yanlış yazılanlar var. Bunlar başarımı dşrecektir?
         1. (Ahmet Çakar (d. 1962, Çankırı, Türkiye)sonuç-->(İstanbul))
3. Hatalara bakılacak.
4. Gerekiyorsa templatelerde düzenleme yapılacak.
5. Gerekiyorsa yeni templateler tanımlanacak.
6. DB deki aşağıdaki alanlar için tum işlemler tekrarlı yapılacak.
   1. isim
   2. doğum\_
   3. doğum\_yeri
   4. ölüm\_tarihi
   5. ölüm\_yeri
   6. diğer\_adları
   7. meslek
7. İl ilçe gibi bilgilerde templatelerde çıkartılacak.
8. Literatur çıkartımı.
9. 550 den örnek alıp bunların ne kadarının gerçekte olmadığının bulunması.
   1. Bu işlem sırasında infoda olup, textte yoksa bunları önce eleyebiliriz.
10. Link olayını incele

Yapılanlar:

1. DB de doğum yeri alanı ayrıştırıldı.
2. Templateler için örnek cümleler çıkartıldı.( TemplateOrnekDoğumYeri.docx)(02.05.2012)
3. Templateler çıkartılacak .( TemplateOrnekDoğumYeri.docx (02.05.2012)
4. Tum templateler implement edildi. (13.05.2012)
5. Bulunan keliemlerin ekleri çıkartıldı. (13.05.2012)
6. Bulunan kelimelerin sayı olamaması gerek bu kontrol edildi. (13.05.2012)
7. Eğer bulunan yer ismi “ili” içeriyorsa atılıyordu. Ancak yanlış sonuçlar rettiğinden burası değiştirildi.”ilini” yapıldı.
8. Bulunan değerlerin veritabanına performanslı bir şekilde yazdırılması yapıldı.
9. Sonuçlar:
   1. Toplam 2400 kayıt var.
   2. 619 tanesinin infobox da doğum yeri yok

( select count(\*) from infobox ib where ib.p\_dogum\_yer = "null" or ib.p\_dogum\_yer is null)

* 1. Kalan kayıt 1780
  2. 905 doğru bulunan
  3. 885 yanlış bulunan
     1. 935 tanesini boş (Bunların 388 tanesi zaten yok infodada boş)
     2. İnfoda dolu olup benim bulamaddığım yada yanlış bulduğum=550 tane

1. 550 adet hiç bulunamayana bakılacak.
   1. Splitlerin yerlerini tespit ederken bazı problemler va. Sıra ile değilde önce hangisi bulunursa şeklinde değiştirildi.
   2. Doğru bulunan sayısı 938
   3. Tamamen boş olan sayısı=923 (Bunların 388 tanesi zaten yok infodada boş)
   4. İnfoda dolu olup benim bulamadığım yada yanlış bulduğum=526 tane
   5. Bulduğum ve eşleşmeyen hiç yok
2. 526 adet bulunamayana bakılacak
   1. Doğru bulunan 951+224?
   2. Boşlukla ilgili bir problem var . Duzenlendi 510 a duştu
   3. Bunlardan çoğunda text içerisinde doğum tarihi bulunmuyor.
   4. İnfoda değeri olmayıp benim bulduğum ise 224 adet
3. 510 adet bulunamayana bakılacak
   1. Kişi infosu olup kişi olmayanlar var. 9063527

(27.05.2012)

1. Makale Okundu.
   1. <http://www.hpi.uni-potsdam.de/fileadmin/hpi/FG_Naumann/publications/2010/iPopulator-CIKM.pdf>
      1. <http://www.hpi.uni-potsdam.de/naumann/projekte/completed_projects/ipopulator.html>
   2. İngilizce için 2010 yılında yapılmış
   3. CRF kullanmışlar.
      1. As CRF implementation, we used CRFsuite [16] with L-BFGS [5] as

feature weight estimation method.

16. N. Okazaki. CRFsuite: a fast implementation of Conditional Random Fields

(CRFs), 2007. <http://www.chokkan.org/software/crfsuite/>.

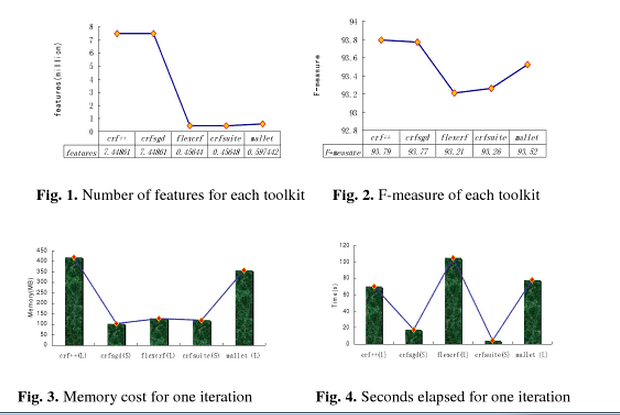
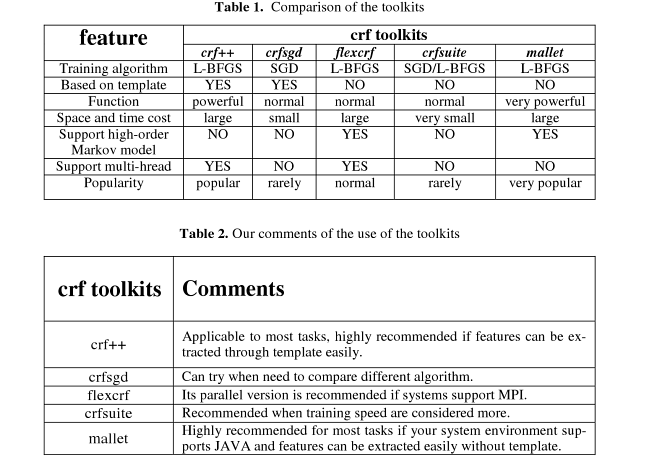
5. R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A Limited Memory Algorithm for

Bound Constrained Optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, 1995.

* 1. Bizim gittiğimiz yoldan gitmişler. En sonunda CRF ile beslemişler.

1. Özet

**ABSTRACT** Roughly every third Wikipedia article contains an infobox - a table that displays important facts about the subject in attribute-value form. The schema of an infobox, i.e., the attributes that can be expressed for a concept, is defined by an infobox template. Often, authors do not specify all template attributes, resulting in incomplete infoboxes. With iPopulator, we introduce a system that automatically populates infoboxes of Wikipedia articles by extracting attribute values from the article's text. In contrast to prior work, iPopulator detects and exploits the structure of attribute values to independently extract value parts. We have tested iPopulator on the entire set of infobox templates and provide a detailed analysis of its effectiveness. For instance, we achieve an average extraction precision of 91% for 1,727 distinct infobox template attributes.

1. CRF ++ incelendi.(2005)
   1. C++ tabanlı bir uygulama
   2. Dosyalar ile çalışıyor.
   3. Templateler verilebiliyor.Template dosyaları hazırlanıyor(Bu biraz karışık)
   4. Verilen textlerde train dosyalrı gibi parse edilerek future setler ile veriliyor.
   5. CRF ++ ‘ verilen dosyadan sonra labellı hali geri dönüyor.
   6. Multithread destekliyor.
2. Crfsgd inceleniyor(2006)
   1. C++ ile yazılmış.
   2. Feature templateler kullanıyor.
3. Flexcrf inceleniyor.(2005)
   1. C++ ile yazılmış.
   2. First order ve second order markov crf destekliyor.?
   3. Paralel çalışabiliyor.
4. Crfsuite inceleniyor(2007)
   1. Hızlı olması için C de yazılmış.
5. Mallet inceleniyor.
   1. Java ile yazılmış.
   2. Birçok NLP işlemi için kullanılıyor.
   3. High level Markov orderları destekliyor.
   4. En iyi sonuçları dönurelnlerden.
6. Mallet inceleniyor…
7. CRF toollarının karşılaştırması yapılıyor.
   1. <http://link.springer.com/chapter/10.1007%2F978-3-642-14831-6_26#page-1>
   2. <http://books.google.com.tr/books?id=KBOIX8TnWgoC&pg=PA197&dq=%22CRFsuite%22+or+mallet&hl=tr&sa=X&ei=c2qiUd6lEoeHhQeqzYHIBA&ved=0CDEQ6AEwAA#v=onepage&q&f=false> (Bunun tamamını iş yerinden indirmek lazım)

Başaramadıkları:

jan legindork