Interacting Croatian NERC system and Intex/NooJ environment

Božo Bekavac, Željko Agić, Marko Tadić

University of Zagreb,
Faculty of Humanities and Social Sciences
{bbekavac, zagic, marko.tadic}@ffzg.hr

NooJ2008
Budapest
2008-06-09
Talk outline

- motivation
  - large scale NLP systems
  - cooperation of linguists and engineers
- proposed solution
  - system design in Intex/NooJ
  - utilization via programming library
- implementation
- concluding remarks
Motivation - 1/3

- large scale NLP systems being developed for Croatian
  - document classification and indexing
  - information retrieval in general
  - developed by a team of linguists and computer engineers
    - different communities – different approaches
    - precise linguistic description vs. approximations
    - usage of lemmatization vs. stemming
Motivation - 2/3

- **user requirements**
  - precision and recall as high as possible
  - speed and robustness

- **our approach**
  - best of both worlds
    - computational linguistic preprocessing & document classification
    - classification favours machine learning methods
    - feature selection: language-specific (incl. NE)
    - **internal requirement**: modular applications i.e. compatible modules in already developed common library (TMT)
Motivation - 3/3

- procedures for feature selection
  - MSD tagging
  - lemmatization
  - named entity recognition and classification

- our share of work
  - PoS/MSD-tagger & lemmatizer had to be adapted as modules
  - NERC system previously developed in Intex

*remember internal project requirements?*
Proposed solution - 1/3

- Croatian linguists use Intex/NooJ
  - named-entity recognition (Bekavac 2005)
  - chunking (Vučković, Tadić, Dovedan, LREC2008)
  - shallow parsing (Vučković, Mikelić-Preradović, Dovedan, NooJ2008)
  - deep(er) parsing in perspective

- why should we stick to it further?
  - fast and elegant system design for development and testing phase
  - capturing knowledge of linguists
    
    *They rarely write it down in standard C++.*
Proposed solution - 2/3

- however...
  - black-box modules and APIs are required
    *You don’t have it if you don’t make a library out of it!*

- our proposed solution
  - use already developed resources for Croatian in Intex/NooJ environment
  - export them and feed them to a black box module, capable of applying it on Croatian texts
Proposed solution - 3/3

- pros < cons
  - easier integration in broader NLP systems
  - easier to modify for needs mentioned
  - lack of Intex/NooJ specifics leads to lower P&R
  - time consuming development and editing
  - language resources created for Intex/NooJ

- however...
  - we have a requirement that needs to be fulfilled
    *User and system requirements say so!*
Implementation - 1/4

- requirements are set by the classification module
  - object-oriented programming library
  - standard ISO/IEC C++ (*wishful thinking*)
  - basic input
    - a sentence written in Croatian
    - rules created by experts using Intex/NooJ
  - required output
    - whatever is defined by the rules
    - primary focus is NERC (for the time being)
• analysis and design
  • input sentence is a `std::vector<std::string>`
    
    *No discussion allowed there.*

  • input rules of Croatian NERC are regular pattern detectors (FST) that operate on sentence level

  • Intex delivers resources
    • GRF and FST files
    • export (Non-)Deterministic C Transition Table
      
      *Most useful.*
what needed to be done?
- non-standard object-oriented FST interface
- building FSTs from Intex/NooJ exports
- running FSTs on single sentences of Croatian

we chose the hard way
- there are other FST libraries out there
  - OpenFST, SFST, libFSM, …
- learn from them, develop your own
  - flexibility regarding special requirements
Implementation - 4/4

- **system specifics**
  - sentences are tagged and lemmatized
    - trigram tagging, Croatian morphological lexicon
    - no ambiguity; could be used in NE normalization
  - special lexicons are applied
    - detailed lists of Croatian locations, organizations
    - project-driven lexicon assembly (newspaper texts)
  - rules are applied sequentially
    - as defined in (Bekavac 2005)
Instead of evaluation

- system still under development
  - ETA 2008-06-20
  - What would we provide anyway?
- evaluation metrics
  - rules are deterministic
    *If the engine works, it works as Intex does.*
  - processing speed comparison?
    *Our system is not created to compete, but we should evaluate it nevertheless.*
Concluding remarks - 1/2

- **applications**
  - every system developed in Intex/NooJ becomes available as a program module
    
    *rule-based tagger, chunker, parser...*

- **future work**
  - complete what we started
  - going above sentence level
  - generating specialized NE lexicons
  - named entity normalization
  - fine-tuning rules of (Bekavac 2005)
Concluding remarks - 2/2

- connecting NooJ with other communities
  - connecting with MulText(East) community
    - Vitas & Erjavec 2004
    - a tool we have presented in Barcelona 2007
    - via conversion of generated inflectional lexicons
  - other FST communities
    - similar to what Kimmo presented this morning
Interacting Croatian NERC system and Intex/NooJ environment

Božo Bekavac, Željko Agić, Marko Tadić

University of Zagreb, Faculty of Humanities and Social Sciences
{bbekavac, zagic, marko.tadic}@ffzg.hr

NooJ2008
Budapest
2008-06-09