The Ninth Conference
On Language Engineering
December 23-24, 2009, Cairo, Egypt

Organized by

Egyptian Society of Language Engineering (ESOLE)

Under the Auspices of

PROF. DR. HANY HELAL
Minister of Higher Education and Scientific Research

PROF. DR. TAREK KAMEL
Minister of Communications and Information

PROF. DR. YOSSRI ELGAMAL
Minister of Education

PROF. DR. AHMAD ZAKI BADR
President of Ain Shams University

PROF.DR. HADIA ELHENAWY
Dean, Faculty of Engineering, Ain Shams University

CONFERENCE CHAIRPERSON
PROF. DR. M. A. R. GHONAIMY

CONFERENCE COCHAIRPERSON
PROF. DR. SALWA ELRAMLY

Faculty of Engineering –Ain Shams University

http: //www.esole.org

Conference Chairman : **Conference Sponsors**

Prof. Dr. M.A Ghonaimy

Technical Program Committee:

Prof. Taghrid Anber , **Egypt**
Prof. I. Abdel Ghaffar , **Egypt**
Prof. M. Ghaly, **Egypt**
Prof. M. Z. Abdel Mageed, **Egypt**
Prof. Khalid Choukri, ELDA, **France**
Prof. Christopher Ciri, LDC, **U.S.A**
Prof. Mona T. Diab, Stanford U., **U.S.A**
Prof. Ayman ElDossouki, **Egypt**
Prof. Afaf AbdelFattah, **Egypt**
Prof. Y. ElGamal**, Egypt**
Prof. M. Elhamalaway**, Egypt**
Prof. S. Elramly, **Egypt**
Prof. H. Elshishiny**, Egypt**
Prof. A. A. Fahmy**, Egypt**
Prof. I. Farag, **Egypt**
Prof. Magdi Fikry, **Egypt**
Prof. Wafa Kamel, **Egypt**
Prof. S. Krauwer, **Netherlands**
Prof. Bente Maegaard, CST, **Denmark**
Prof. A. H. Moussa, **Egypt**
Prof. M. Nagy**, Egypt**
Prof. A. Rafae, **Egypt**
Prof. Mohsen Rashwan**, Egypt**
Prof. H.I. Shaheen**, Egypt**
Prof. S.I. Shaheen**, Egypt**
Prof. Hassanin M. AL-Barhamtoshy**, Egypt**
Prof. M. F. Tolba, **Egypt**
Dr. Tarik F. Himdi**, Saudi Arabia**

Organizing Committee

Prof. I. Farag
Prof. Taghrid Anbar
Prof. H. Shahein
Eng. Mona Zakaria
Prof. S. Elramly
Prof. Hany Kamal
Dr. A. Bahaa
Eng. Bassant A. Hamid

Conference Secretary General

Prof. Dr. Salwa Elramly

The Ninth Conference on Language Engineering

Final Program

Wednesday 23 December 2009

9.00 - 10.00 Registration

10.00 - 10.30 Opening Session

10.30 - 11.00 **Session 1** : **Invited Paper 1**:
Chairman : Prof. Dr. Ibrahim Farag

صناعة المحتوى العربية : رؤية مستقبلية
د.نبيل على
خبير فى بحوث اللغويات الحاسوبية

11.00 - 12.00 Coffee break

12.00 - 12.30 **Session 2** : **Panel Session:**
Chairman : Prof. Dr. Ibrahim Farag
New Trends in Machine Learning (Deep Learning)
Mohsen A.Rashwan
Professor, Electronics and Electrical Communications Department, Faculty of Engineering, Cairo University, Cairo, Egypt

12.30 - 13.00 **Session 3**: **Invited Paper 2:**
Chairman : Prof. Dr. Adeeb Riad Ghonaimy
Arabic Optical Character Recognition: State of the Art
Mohammed F. Tolba
Professor, Department of Scientific Computing, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt

13.00 - 14.00 **Session 4**: **Machine Translation:**
Chairman : Prof. Dr. M. Fahmy Tolba

1. **The Universal Networking Language in Action in English-Arabic Machine Translation**
   Sameh Alansary[1], Magdy Nagi[2], Noha Adly[2]
   [1]Department of Phonetics and Linguistics, Faculty of Arts, Alexandria University, Alexandria, Egypt
   [2]Computer and System Engineering Dept, Faculty of Engineering, Alexandria University, Egypt
2. **Issues on Interlingua Machine Translation Systems**
   Sameh Alansary
   Department of Phonetics and Linguistics, Faculty of Arts, Alexandria University, Alexandria, Egypt

14.00 - 15.00 Lunch

15.00 - 17.00 **Session 5** : **Room A** : **Language Analysis and Comprehension**
Chairman : Prof. Dr. Aly Aly Fahmy

1. **Bootstrapping a Lexicon-free Tagger for Arabic**
   Allan Ramsay & Yasser Sabtan

School of Computer Science, University of Manchester, UK

2. **A Ranking approach for Arabic Root Extraction using Machine Readable Dictionaries**
   Soha M. Eid[1], Nayer M. Wanas[2], Nadia H. Hegazy[3] , Mohsen A. Rashwan[4]
   [1]Assistant Researcher, Informatics Department, Electronics Research Institute, Cairo, Egypt
   [2]Assistant Professor, Informatics Department, Electronics Research Institute, Cairo, Egypt
   [3] Professor, Informatics Department, Electronics Research Institute, Cairo, Egypt
   [4]Professor, Electronics and Electrical Communications Department, Faculty of Engineering, Cairo University, Cairo, Egypt
3. **Automated Free-Text Answers Assessment**
   Talal Saeed Saleh[1], Ahmed Hussein Kamal[1], Ali Ali Fahmi[1]
   Computer Science Dept, Faculty of Computers and Informatics, Cairo University
4. **Stem-Based vs. Word-Based Language Models For The Modern Standard Arabic (MSA)**
   Mohsen Moftah[1] , Waleed Fakhr[1] , Sherif Abdou[2] , Mohsen Rashwan[3]
   [1] Arab Academy for Science, Technology and Maritime Transport
   [2] Faculty of Computers and Information Systems, Cairo University ,Giza, Egypt
   [3] Faculty of Engineering, Cairo University, Giza, Egypt

15.00 - 17.00 **Session 6** : **Room B** : **Language Generation**
Chairman Prof. Dr. Mohsen Rashwan

1. **The /nafs-/ Construction in Arabic**
   Prof. Huda M. M. Ghaly
   Theoretical linguistics at English Dept, Faculty of Arts, Ain Shams University
2. **A Semantic Graph Reduction Approach for Abstractive Text Summarization**
   Ibrahim F. Moawad, Mostafa M. Aref
   Faculty of Computer and Information Sciences, Ain-Shams University
3. **التصنيف الآلي للنصوص العربية**
   سيد محمد سيد
   شركة حرف لتقنية المعلومات

Thursday 24 December 2009

10.00 - 11.30 **Session 7** : **Room A: Natural Language Processing for Information Retrieval**

Chairman : Prof. Dr. Hassanin M. Al-Barhamtoshy

1. **Lexical and Morphological Statistics of an Arabic POS-Tagged Corpus**
   Hamdy S. Mubarak, Kareem A. Shaban, Forat M. Adel
   Arabic NLP Researches, Sakhr Software, Sakhr Building, Cairo, Egypt
2. **Linguistic Resources for English/Arabic CLIR: A Comprehensive Survey**
   Tarek Elghazaly and Aly Aly Fahmy
   Faculty of Computers and Information, Cairo University, Giza, Egypt
3. **An Empirical Analysis of Lexical Text Similarity**
   Abdulrahman G. Abdulwahab, Ibrahim F. Imam
   Computer Science Department, Arab Academy for Science & Technology & Maritime Transport, Cairo, Egypt

10.00 - 11.30 **Session 8** : **Room B**: **Speech Processing, Recognition and Synthesis**
Chairman : Prof. Dr. Waleed Fakhr

1. **Speech Recognition of Arabic Digits Using G.729 Coder and GSM Platform Simulator for Mobile systems**
   Nariman A. El-Salam[1] , Neamat A. El-Kader[2], Mona M. Reiadh[2]
   [1] Assistant lecturer in Department of Electrical Engineering, Modern Academy in Cairo, Egypt
   [2] Professor in Department of Electrical Engineering, Cairo University, Egypt
2. **Speech Processing Framework: the Generic Speech Library (GSL)**
   Amr M. Gody
   Fayoum University, Egypt
3. **Designing and Implementing Arabic Text - to - Speech**
   Hassanin M. Al-Barhamtoshy and Wajdi H. Al-Jideebi
   Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia

11.30 - 12.00 Coffee Break

12.00 - 12.30 **Session 9** : **Room A** : **Invited Paper 3**
Chairman : Prof. Dr. Younis Elhamalawy

المعجمية العربية الحاسوبية
أ. د. وفاء كامل فايد
أستاذة اللغويات – كلية الآداب، جامعة القاهرة

12.30 - 13.00 **Session 10**: **Room A: Evaluation of Natural Language Processing Systems**
Chairman : Prof. Dr. Younis Elhamalawy

مقترح لمعايير وضوابط تقييم المحلِّلات الصرفية
د. سلوى حماده
معهد بحوث الإلكترونيات وخبيرة المنظمة العربية للتربية والعلوم والثقافة

12.30 - 13.00 **Session 11: Room B: Automatic Character Recognition**

Chairman : Prof. Dr. Salwa El Ramly

تطوير التعرف الآلي على الحروف العربية من خلال أداة لغوية غير صرفية

عمرو جمعة عبد الرسول

شركة حرف لتقنية المعلومات

13.00 - 14.00 **Session 12: Room A: Semantic Web and Ontology Languages**

Chairman : Prof. Dr. M.Zaki Abdel Mageed

1. **Designing and Implementing Arabic WordNet Semantic-Based**
   Hassanin M. Al-Barhamtoshy and Wajdi H. Al-Jideebi
   Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia
2. **Semantic Mediation Between Two Ontologies**
   T. Hossam, M. Zaki
   Computer Engineering Department, Al Azhar University, Egypt

13.00 - 14.00 **Session 13: Room B: Spoken Language Understanding**

Chairman : Prof. Dr. Hany Mahdy

1. **Arabic Speech Keyword Spotting Through IP-Based Networks with Packet Loss**
   M. Hesham, and M. Osama
   Faculty of Engineering, Cairo University, Egypt
2. **Optimization Techniques for Speech Emotion Recognition**
   Julia Sidorova
   Universitat Pompeu Fabra, Barcelona, Spain

14.00 - 14.30 **Session 14 : Room A : Invited paper 4**

Prof. Dr. Younis Elhamalawy

An Overview of Tied-mixture Language Models

Mohamed Afify[1], Ruhi Sarikaya[2]

[1] Orange Labs, Cairo, Egypt

[2]IBM T.J. Watson Research Center, Yorktown Heights, NY, USA

14.30 - 15.30 Lunch

15.30 - 16.30 **Session 15 : Panel Discussion : Room A :** Language Engineering Serving Persons With Special Needs

Chairman : Prof. Dr. Hassanin M. Al-Barhamtoshy

16.30 - 17:00 Closing session

أعضاء الجمعية من المؤسسات

1- مركز نظم المعلومات – كلية الهندسة - جامعة عين شمس

2- معهد الدراسات والبحوث الإحصائية - جامعة القاهرة
3- مركز الحساب العلمى - جامعة عين شمس
4- الأكاديمية العربية للعلوم والتكنولوجيا والنقل البحرى
5- أكاديمية أخبار اليوم
6- معهد بحوث الإلكترونيات
7- معهد تكنولوجيا المعلومات
8- مكتبة الإسكندرية
9- المعهد القومى للاتصالات (NTI)
10- الشركة الهندسية لتطوير نظم الحاسبات (RDI)
11- الهيئة القومية للاستشعار من بعد و علوم الفضاء
12- كلية الحاسبات و المعلومات جامعة قناة السويس
13- دار التأصيل للبحث و الترجمة

أهداف الجمعية

1- الاهتمام بمجال هندسة اللغويات مع التركيز على اللغة العربية بصفتها لغتنا القومية والتركيز على قواعد البيانات المعجمية وصرفها ونحوها ودلالتها بهدف الوصول إلى أنظمة ألية لترجمة النصوص من اللغات الأجنبية إلى اللغة العربية والعكس, وكذلك معالجة اللغة المنطوقة والتعرف عليها وتوليدها, ومعالجة الأنماط مع التركيز على اللغة المكتوبة بهدف إدخالها إلى الأجهزة الرقمية.
2- متابعة التطور فى العلوم والمجالات المختصة بهندسة اللغة
3- التعاون مع الجمعيات العلمية المماثلة على المستوى المحلى والقومى والعالمى.
4- إنشاء قواعد بيانات عن البحوث التى سبق نشرها والنتائج التى تم التوصل إليها فى مجال هندسة اللغة بالإضافة إلى المراجع التى يمكن الرجوع إليها سواء فى اللغة العربية أو اللغات الأخرى.
5- إنشاء مجلة علمية دورية للجمعية ذات مستوى عال لنشر البحوث الخاصة بهندسة اللغة وكذلك بعض النشرات الدورية الإعلامية الأخرى بعد موافقة الجهات المختصة.
6- عقد ندوات لرفع الوعى فى مجال هندسة اللغة
7- تنظيم دورات تدريبية يستعان فيها بالمتخصصين وتتاح لكل من يهمه الموضوع. وذلك من أجل تحسين أداء المشتغلين فى البحث لخلق لغة مشتركة للتفاهم بين الأعضاء
8- إنشاء مكتبة تتاح للمهتمين بالموضوع تشمل المراجع وأدوات البحث من برامج وخلافه.
9- خلق مجال للتعاون وتبادل المعلومات وذلك عن طريق تهيئة الفرصة لعمل بحوث مشتركة بين المشتغلين فى نفس الموضوعات.
10- تقييم المنتجات التجارية أو البحثية والتى تتعرض لعملية ميكنة اللغة.
11- رصد الجوائز التشجيعية للجهود المتميزة فى مجالات هندسة اللغة.
12- إنشاء فروع للجمعية فى المحافظات.

الجمعية المصرية لهندسة اللغة
THE EGYPTIAN SOCIETY OF LANGUAGE ENGINEERING

المؤتمر التاسع لهندسة اللغة

23-24 ديسمبر2009

جمهورية مصر العربية_القاهرة

ينظم المؤتمر

الجمعية المصرية لهندسة اللغة

تحت رعاية

الأستاذ الدكتور/هانى هلال
وزير التعليم العالى والبحث العلمى

الأستاذ الدكتور/ طارق كامل
وزير الاتصالات والمعلومات

الأستاذ الدكتور/ يسرى الجمل
وزير التربية والتعليم

الأستاذ الدكتور/ أحمد زكى بدر

رئيس جامعة عين شمس

الأستاذ الدكتور/هادية سعيد الحناوى
عميد كلية الهندسة - جامعة عين شمس

رئيس المؤتمر
الأستاذ الدكتور/ محمد أديب رياض غنيمى

مقرر المؤتمر
الأستاذ الدكتور / سلوى حسين الرملى
كلية الهندسة - جامعة عين شمس

مكان عقد المؤتمر : كلية الهندسة - جامعة عين شمس
http: // www.esole.org

Issues on Interlingua Machine Translation Systems

The 9th Conference On Language Engineering
23-24 December 2009
Cairo, Egypt

Sameh Alansary
sameh.alansary@bibalex.org

Head of Arabic UNL Language Center
Bibliotheca Alexandrina

Associate Professor, Faculty of Arts,
Alexandria University

Alexandria, Egypt

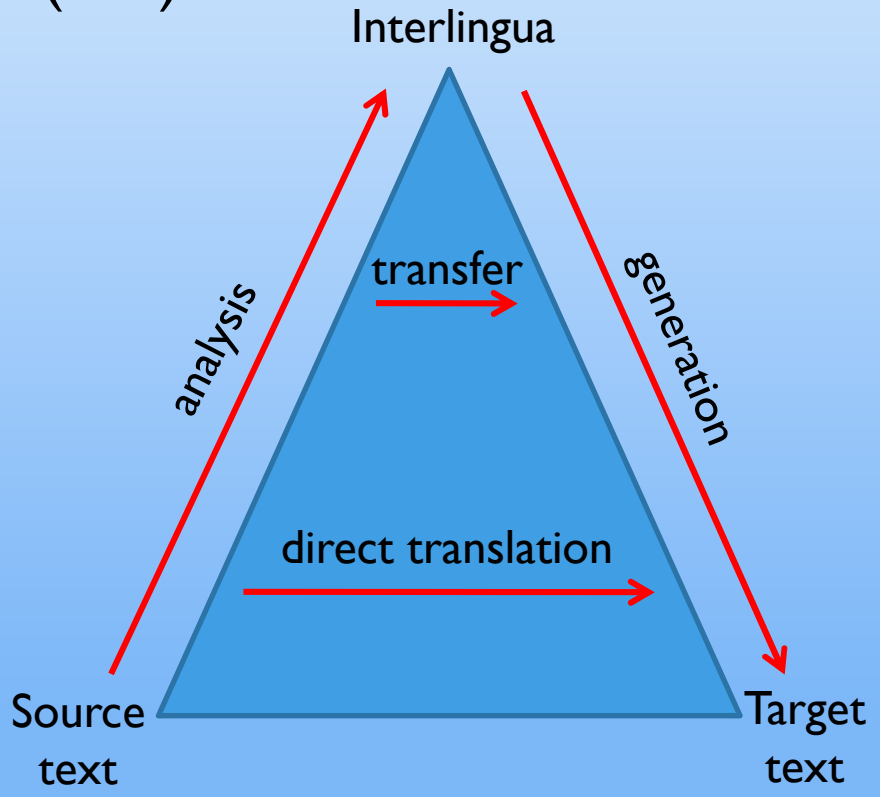
Outline

- Approaches to Machine Translation (MT).
- Interlingua-based MT in details.
- Interlingua MT Systems.
  - UNITRAN
  - KANT
  - DLT
  - UNL

Approaches of Machine Translation

- Machine Translation began in the 1950s based on NLP .
- A number of different approaches have been made to tackle the problem of Machine Translation(MT).

  - Direct approach
  - Transfer approach
  - Interlingua approach



Different Approaches to Machine Translation (MT)

1- Direct Translation Approach.

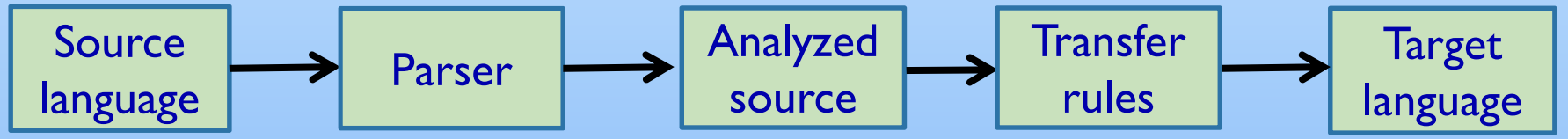
Word-for-word based substitution (with some local adjustment) between language pairs;

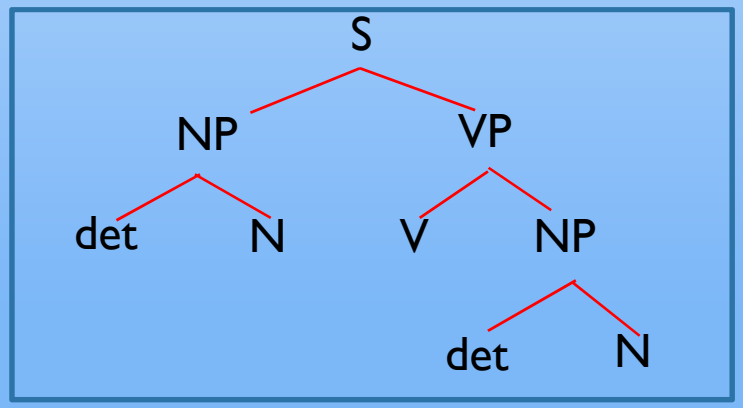
Different Approaches to Machine Translation (MT)

2- Transfer Approach.

Parses source text into a syntactic structure representation, then maps that using transfer rules to a syntactic structure representation of the target language text.

Ex: The boy ate the apple.

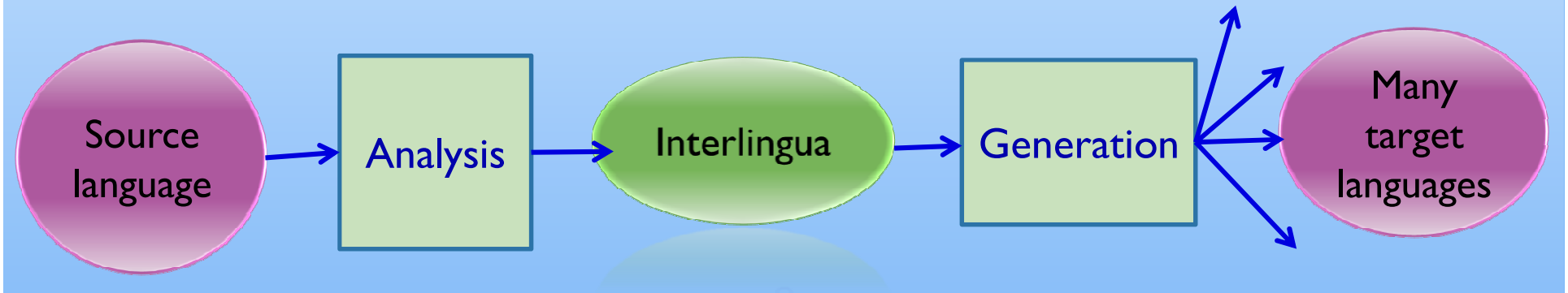




Different Approaches to Machine Translation (MT)

3- Interlingua Approach.

Converts source text into a language neutral, abstract meaning representation, then uses that representation to generate the target text.



BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

Characteristics of an Interlingua

1) Universality.

2) Content rather than Form Representation.

3) Unambiguous.

4) Full Content Representation.

5) Independence of the SL and the TL.

Advantages of Interlingua-based Approach over Other MT Systems

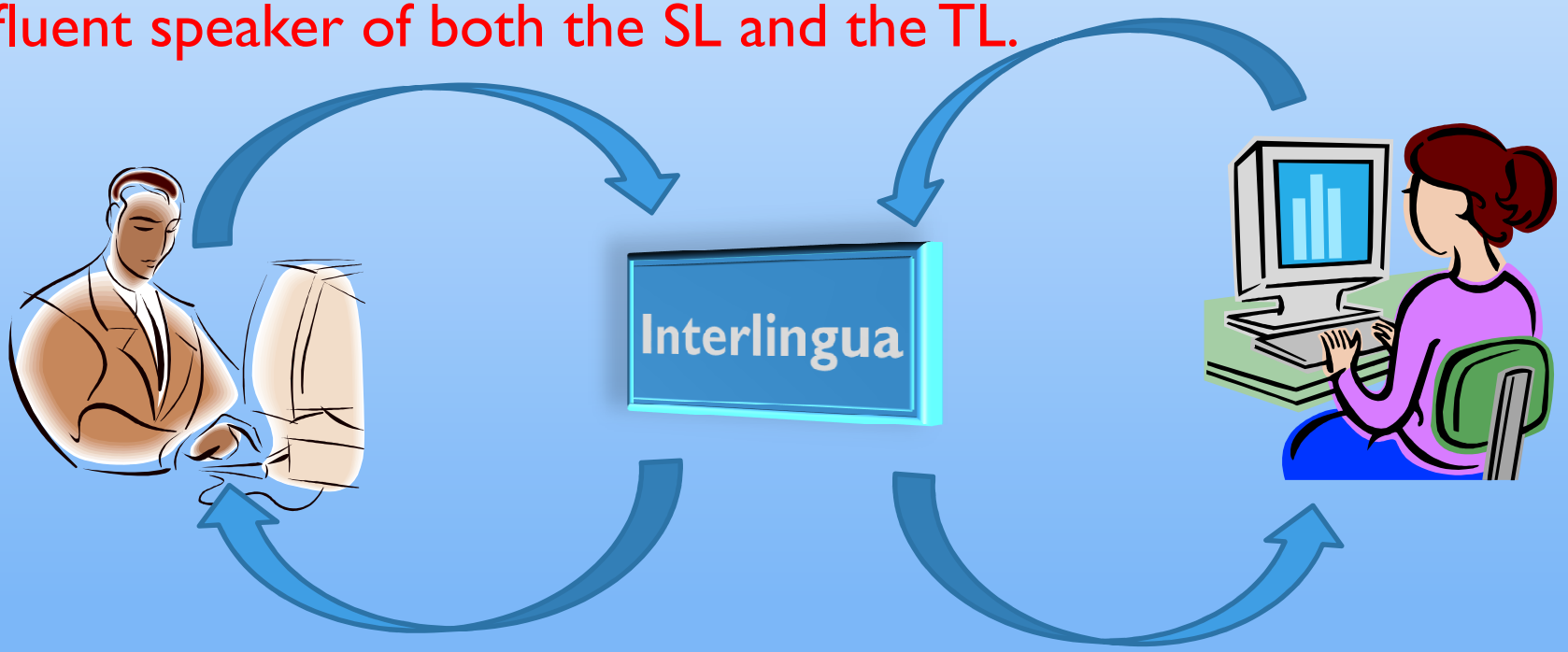
1) Economy in a Multilingual Environment:

- Any number of source and target languages can be connected, without the need to define explicit rules for each language pair in each translation direction.

Advantages of Interlingua-based Approach over Other MT Systems

2) Localization:

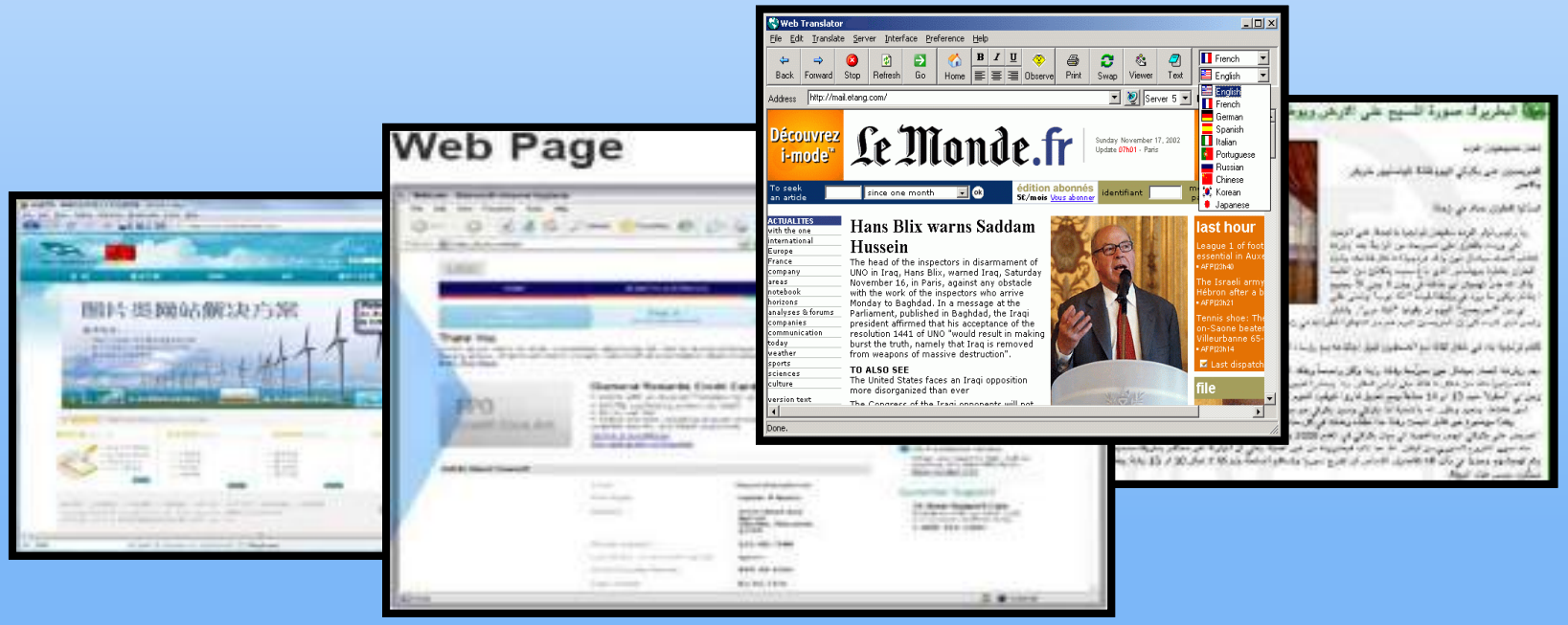
- The development of dictionaries and grammar rules for the analysis and generation of a language from and into and interlingua only requires a well-trained native speaker, rather than a fluent speaker of both the SL and the TL.



Advantages of Interlingua-based Approach over Other MT Systems

3) Instant Translation of Web Information:

If web pages would contain not only the source text but also some interlingua representations thereof, various target-language versions of these web pages can be generated instantaneously, thus, helping the dissemination of knowledge across language barriers.



Advantages of Interlingua-based Approach over Other MT Systems

- An intermediate language-neutral representation of meaning can be used by NLP systems for other multilingual applications such as cross-lingual information retrieval, summarization, and question answering.
- Current systems rely largely on syntactic matching for the gathering of relevant information. Hence, interlingua-based systems can dramatically improve our ability to search for and find information

Challenges facing interlingua-based MT

1) Creating an Independent Language-neutral Representation:

- It is difficult to create an adequate interlingua that is both, abstract and independent of the source and target languages and explicitly preserve the appropriate semantic, pragmatic and other contextual information.

- The more languages added to the translation system, and the more different they are, the more potent the interlingua must be to express all possible translation directions.

2)- Style and Emphasis of SL are Lost

Challenges facing interlingua-based MT

3) divergence:

Categorial Divergence: words in one language may be of a different part of speech in another language.

Example:

The snake is very dangerous ⟵ فالثعبان شديد الخطر

Previously held the post ⟵ سبق أن تولى المنصب

Challenges facing interlingua-based MT

3) divergence:

Conflational Divergence: two or more words in one language may have a one- word counterpart in another.

Example:

يقوم بتغطية ⟹ covers

الاستعانة ب ⟹ seeking the help of

Challenges facing interlingua-based MT

3) divergence:

Structural Divergence: The realization of verb arguments in different syntactic configurations in different languages. For example, *to enter the house — entrar en la casa (enter in the house).*

Example:

وجه الدعوة للرئيس ⟹ Invited the President

أجاب على السؤال ⟹ Answered the question

Interlingua-based MT Systems

UNITRAN :

- The name UNITRAN stands for UNIversal TRANslator, a system that serves as the basis for translation across a variety of languages, not just two languages, or a family of languages.

- Developed by Bonnie Dorr (at Massachusetts Institute of Technology.

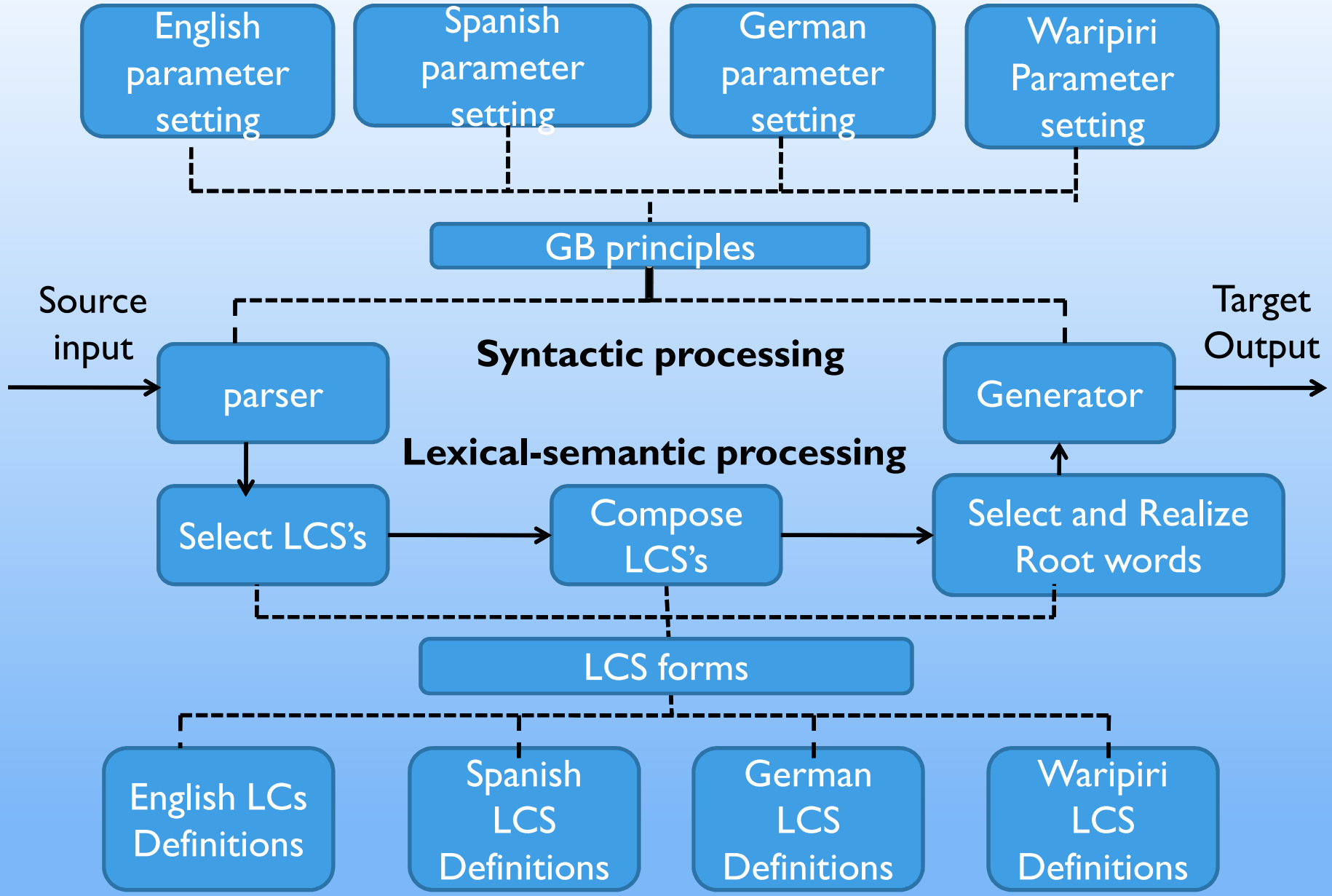
- Currently the UNITRAN system operates bidirectionally between Spanish and English; other languages may easily be added simply by setting the parameters to accommodate those languages.

Interlingua-based MT Systems

- UNITRAN does not incorporate context or domain knowledge, it cannot use discourse, situational expectations, or domain information in order to generate a sentence.

- UNITRAN does not represent the notion of aspect; there is no way to establish whether an event is prolonged, repeated, instantaneous,… etc.

The Architecture of UNITRAN System:



Interlingua-based MT Systems

Kant:

- The KANT project is a part of the Center for Machine Translation (CMT) at Carnegie Mellon University (CMU) and was founded in 1989 for the research and development of large-scale.

- KANT has been applied to the domains of electric power utility management, heavy equipment technical documentation, medical records, car manuals, and TV captions.

- KANT is the only interlingual MT system that has ever been made operational in a commercial setting.

Interlingua-based MT Systems

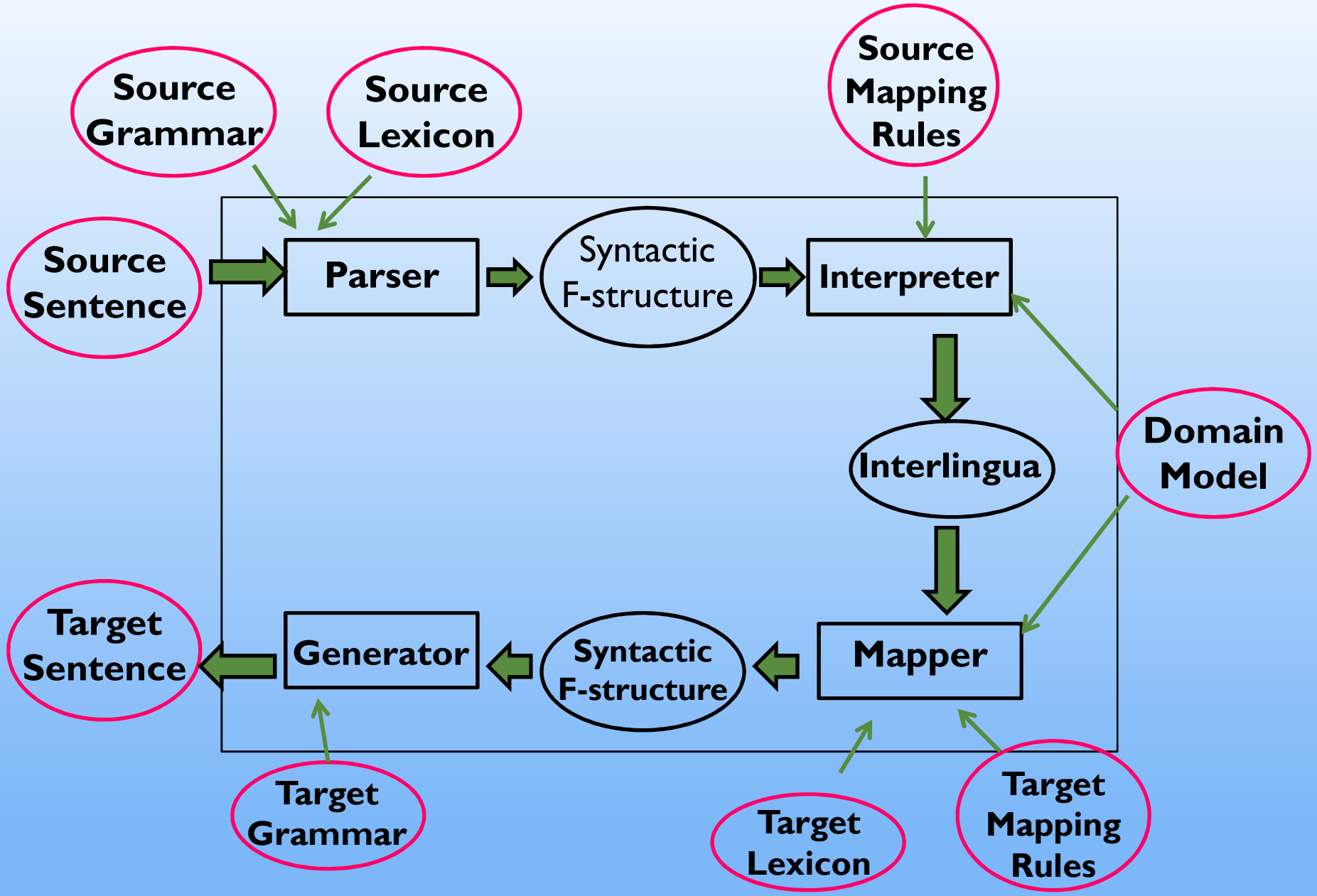
Kant:

- KANT uses a controlled vocabulary and grammar for each source language, and explicit yet focused semantic models for each technical domain to achieve very high accuracy in translation.

- The general (non-domain specific) words used in the source text are limited to a basic vocabulary of about 14,000 distinct word senses. The domain-specific technical terms are limited to a pre-defined vocabulary.

- KANT limits the use of constructions that would create unnecessary ambiguity or other difficulties in parsing.

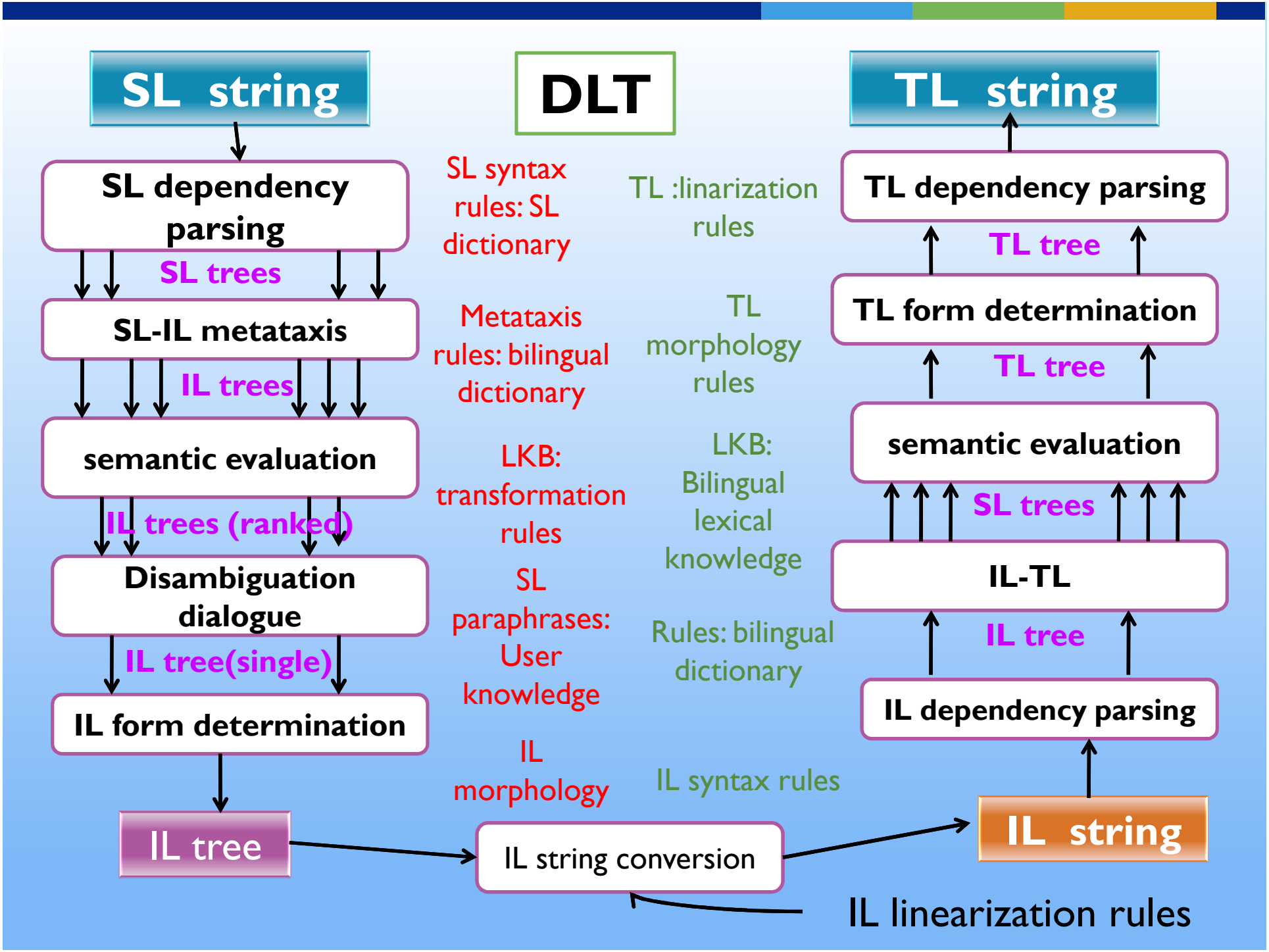
The Architecture of KANT System:



Interlingua-based MT Systems

DLT

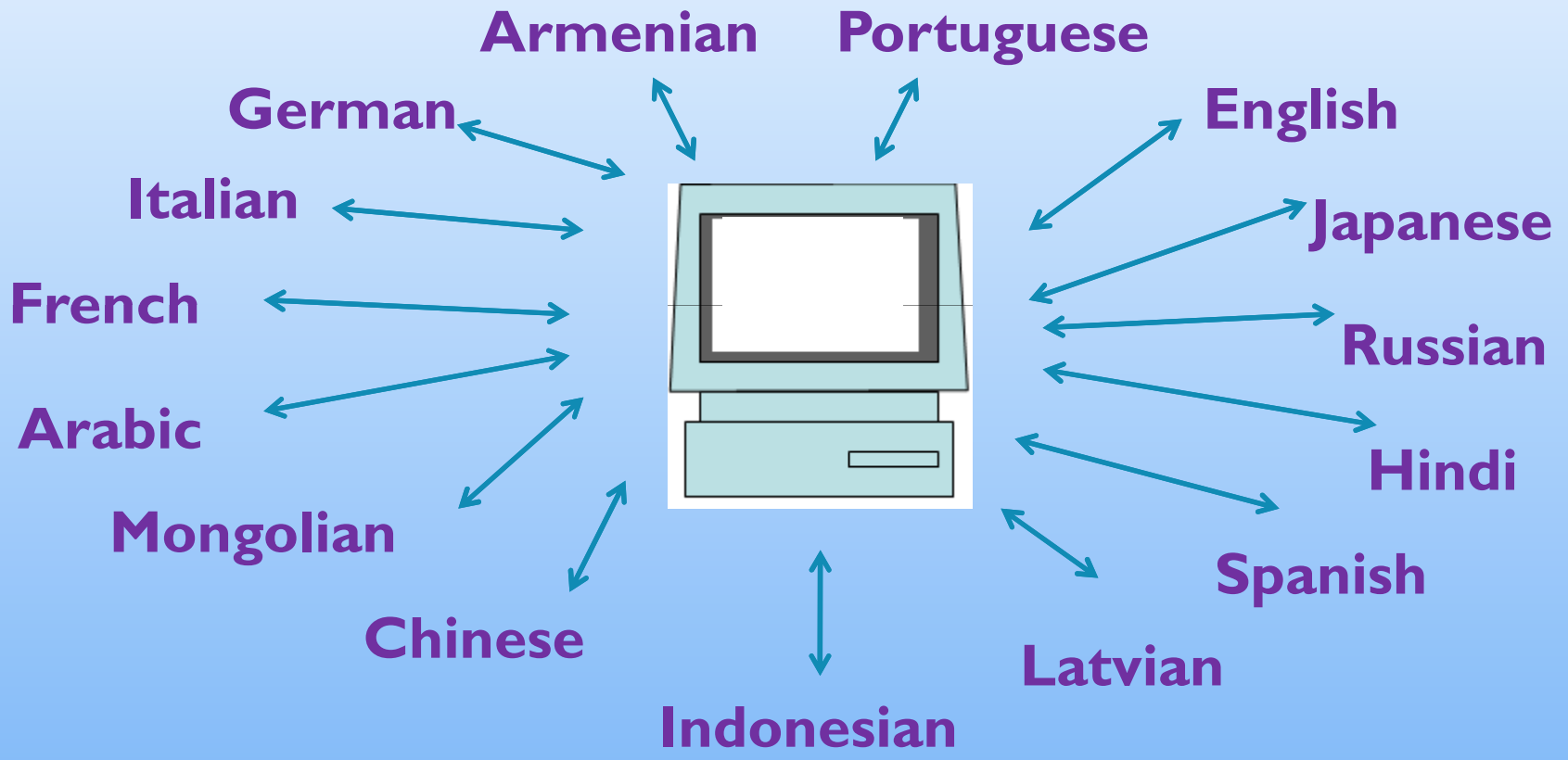
- The DLT (Distributed Language Translation) system was developed at the BSO software company in Utrecht (The Netherlands) was a six-year project from 1985 under the general direction of Toon Witkam.
- DLT was intended as a multilingual interactive system operating over computer networks, designed for monolingual users wishing to convey messages in other languages,
- The system requires interactive collaboration in the analysis and disambiguation of input texts in order that output can be produced fully automatically.
- The distinctive feature of DLT was the use of Esperanto as an intermediate language.


SL string
DLT
TL string
SL dependency parsing
SL trees
SL-IL metataxis
IL trees
semantic evaluation
IL trees (ranked)
Disambiguation dialogue
IL tree(single)
IL form determination
IL tree
SL syntax rules: SL dictionary
Metataxis rules: bilingual dictionary
LKB: transformation rules
SL paraphrases: User knowledge
IL morphology
TL :linarization rules
TL morphology rules
LKB: Bilingual lexical knowledge
Rules: bilingual dictionary
IL syntax rules
IL string conversion
TL dependency parsing
TL tree
TL form determination
TL tree
semantic evaluation
SL trees
IL-TL
IL tree
IL dependency parsing
IL string
IL linearization rules

UNL Interlingua

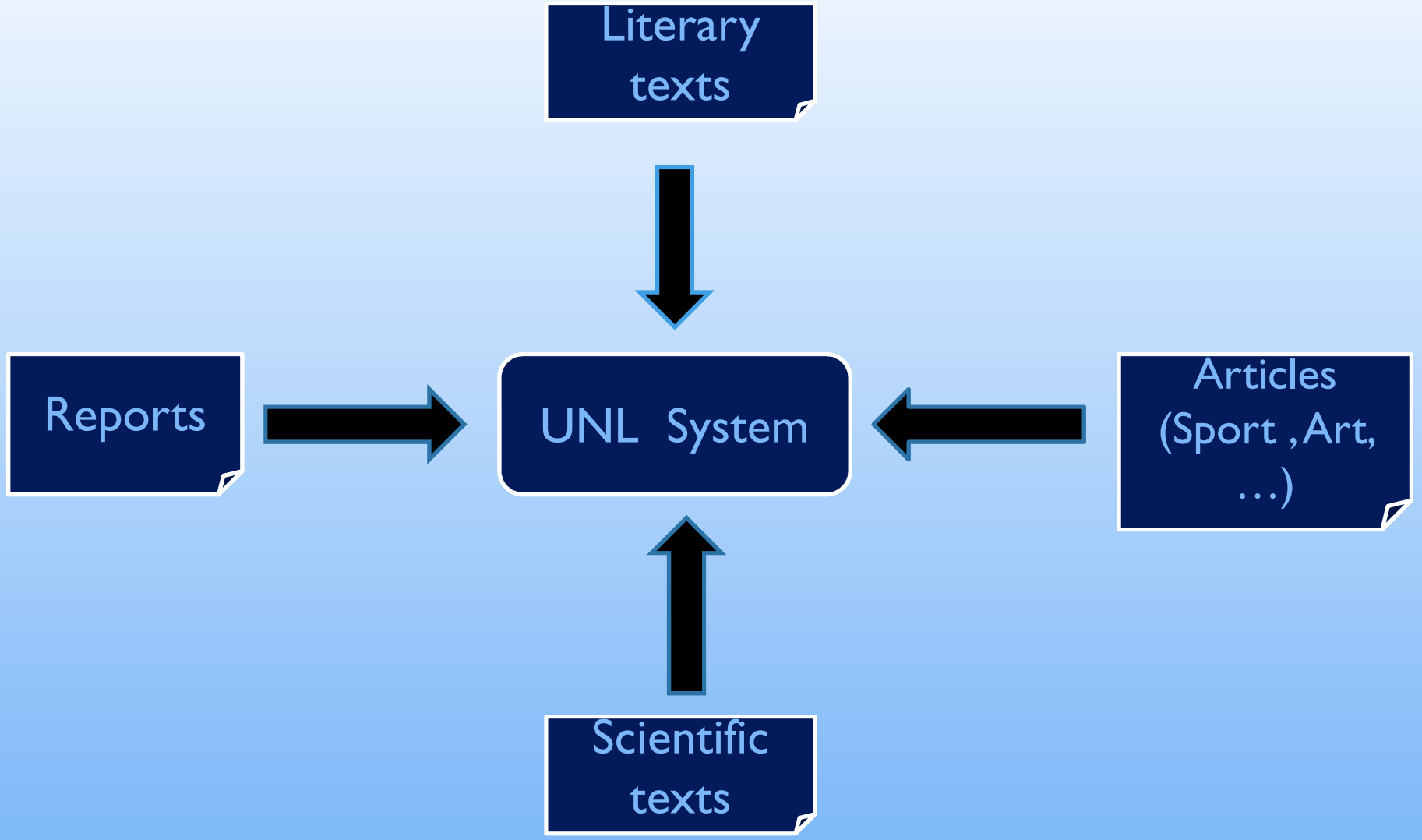
- In 1996 , at the Institute of Advanced- Studies, United Nations University, Tokyo; Japan the Universal Networking Language (UNL) has been developed by H.Ushida as a new interlingual MT system embeddable in html or xml formats for multilingual document representation and processing.

UNL Interlingua



- And since the UNL formal language is put in a universal format so it's easy to transfer from/to any language.

- UNL is not limited to a specific domain.



BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

What is UNL?

Universal

Networking

Language

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

Universal

It represents the meaning of the natural language in a universal format which can work for any natural language.

Networking

It works over the internet.

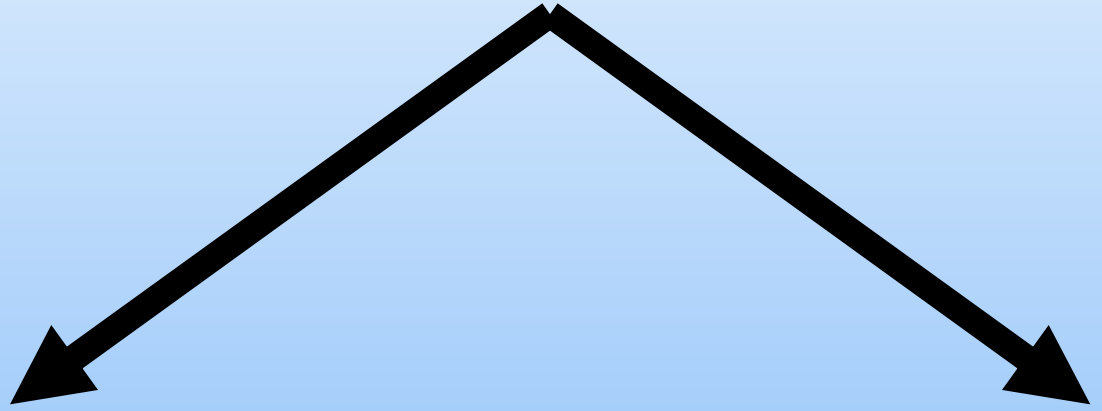
Language

It has all the components corresponding to that of a natural language.

- Vocabulary.
- Syntax.
- Semantics.
- Can express objective and subjective meanings.

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

Universal Networking Language



Formalism

Knowledge representation

System

Knowledge dissemination

UNL as a formal language:
How does it represent knowledge?

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

1- Universal Word

- Words that constitute the vocabulary of UNL.
- Express the meaning of a concept.

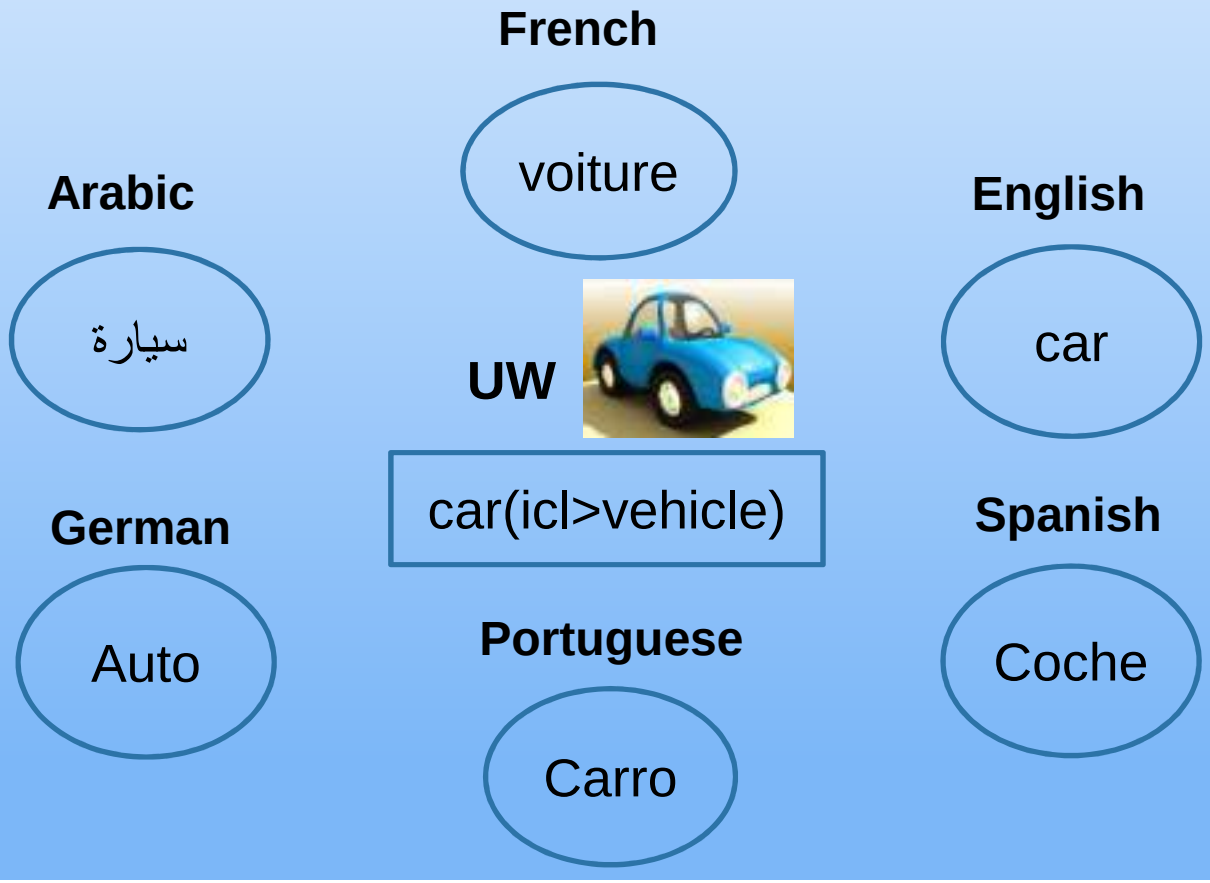
Example: book(icl>document)

book(agt>person,obj>thing))

- A basic element for constructing a UNL expression of a sentence.
- A Universal Word is represented as a node in the semantic network of UNL expression.

Universality of UNL vocabulary

- Every Natural language has its own vocabulary to define the same concept.
- Universal Word in UNL can represent the same concept in different Natural Languages.



Universal Word structure

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

- A UW is made up of a Head and constraints list.

Head

English word, compound word or phrase that is interpreted as a label for a concept.

Constraints list

Restrict the concept of a UW to a subset or to a specific concept and make the concept clear and unambiguous.

Relation Tag

Part of constraints list which determines the relation between the concept and other concepts which exist in the UNL-KB.

icl	iof	equ
author(icl>person)	John(iof> person)	BA(equ>Bibliotheca Alexandrina)

Example

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

The English word state can have several meanings.

A country with its independent government.

state(icl>government)

The government of a country.

state(icl>country)

The mental, emotional or physical condition that person or thing is in.

state(icl>condition)

Express something in words.

state(icl>express(agt>thing,gol>thing,obj>thing))

Basic Categories of UW

- UWs should belong to the following categories.

Nominal concept

pen(icl>tool)	I need my pen

verbal concept

walk(agt>thing)	I walked alone
change(obj>thing)	The weather will change
seem(aoj>thing,obj>thing)	It seems nice

Adjective concept

positive(aoj>thing)	a positive fact
only(mod<thing)	The only person

Adverbial concept

weekly(icl>how)	This class is held weekly.

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

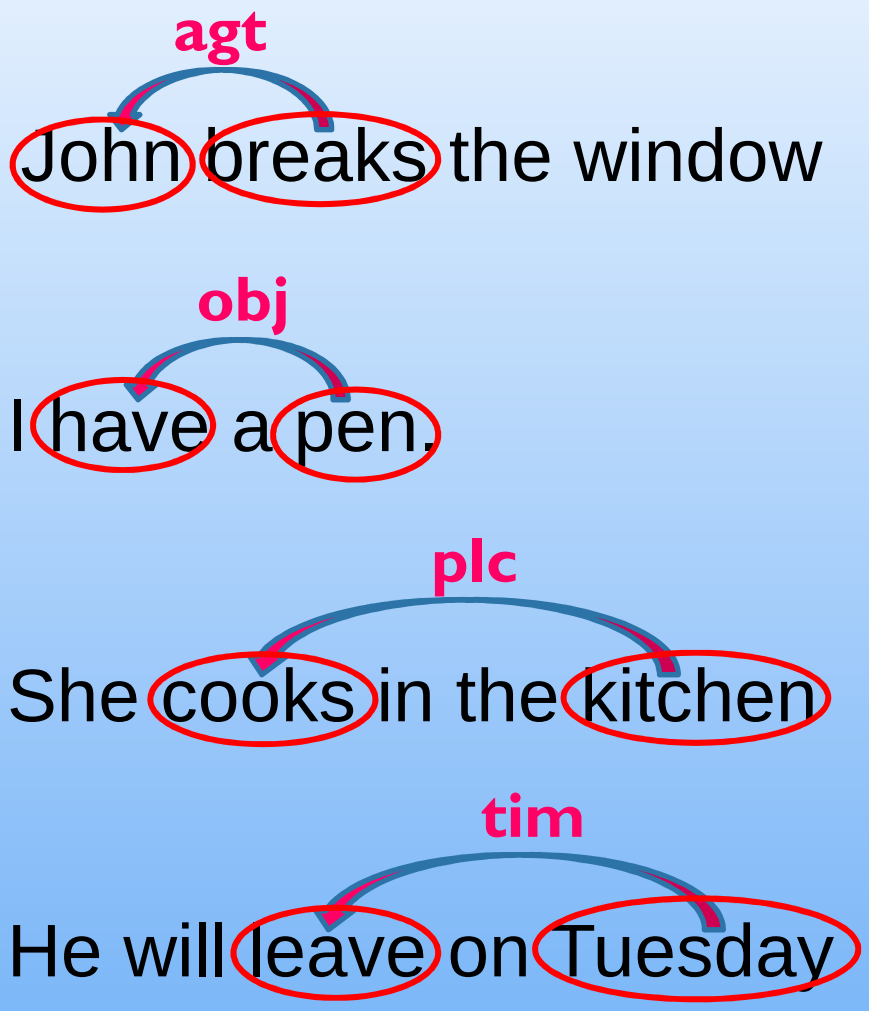
2- Relations

- Constitutes syntax of the UNL.
- Expresses objectivity together with UWs.
- Expresses how concepts(UW) constitutes a sentence related each other.
- They have different labels according to the different role they play.

- 43 semantic relations can be distinguished.

Examples:

agt	agent	John breaks the window
obj	object	I have a pen.
plc	object	She cooks in the kitchen
tim	time	He will leave on Tuesday



3- Attributes

- Express additional information about the universal words appear in a sentence.

Example:

Express subjectivity of the speaker.

@past → He played football.

@progress → I am working now.

Speaker's review of reference to concepts.

@def → the book you lost.

Statement VS Question

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

.@ statement or .@question

Example:

يوجد أحد هناك.
يوجد أحد هناك؟

Polite request

.@polite .@request

Example:

فأعاد علي، بكل هدوء كما لو كان الأمر هاما جدا من فضلك..ارسم لي خروفا

Wishes

.@ wish

Example:

كنت أود لو بدأت قصتي كما تبدأ قصص الجنيات

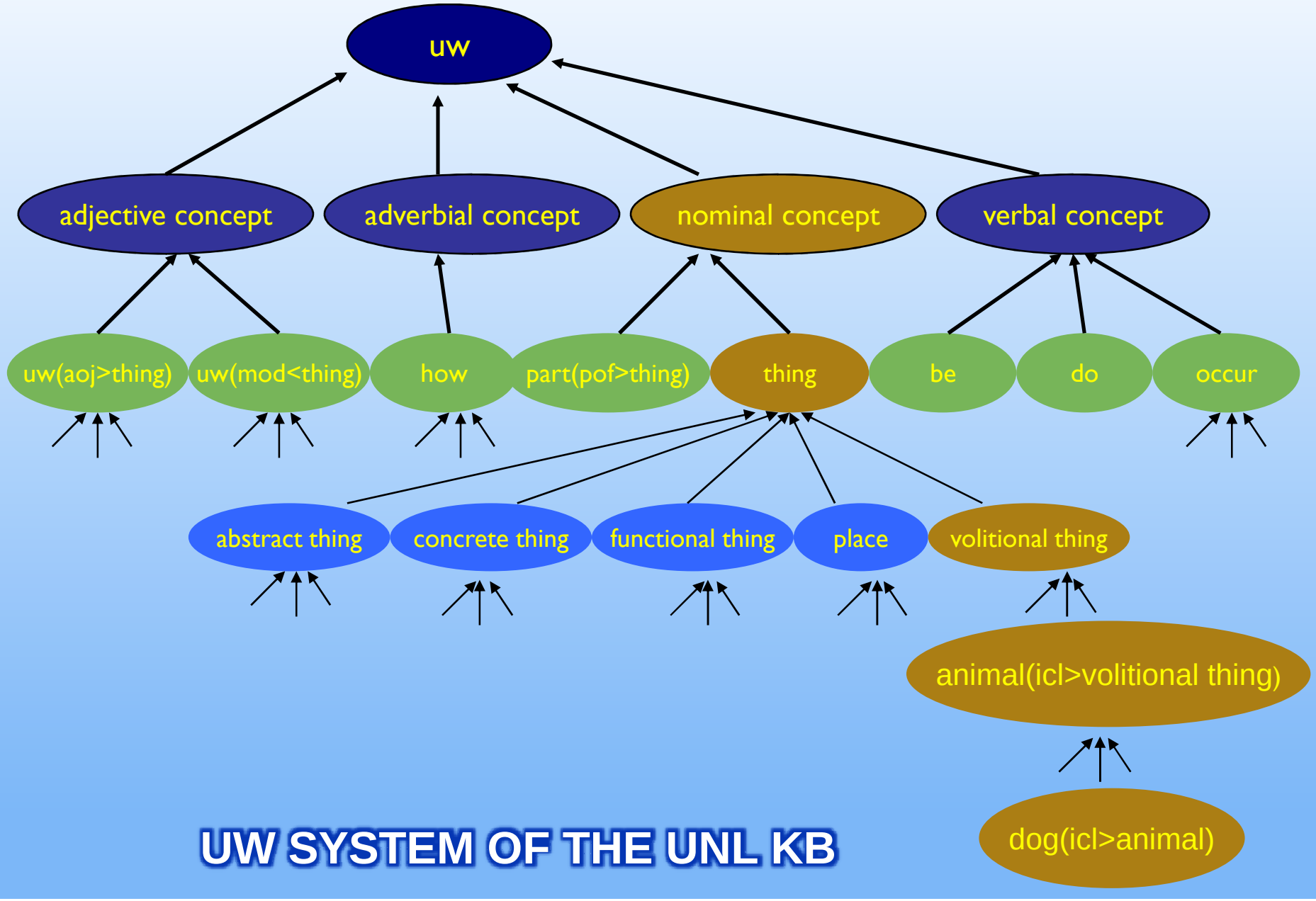
Exclamation

.@exclamation

Example:

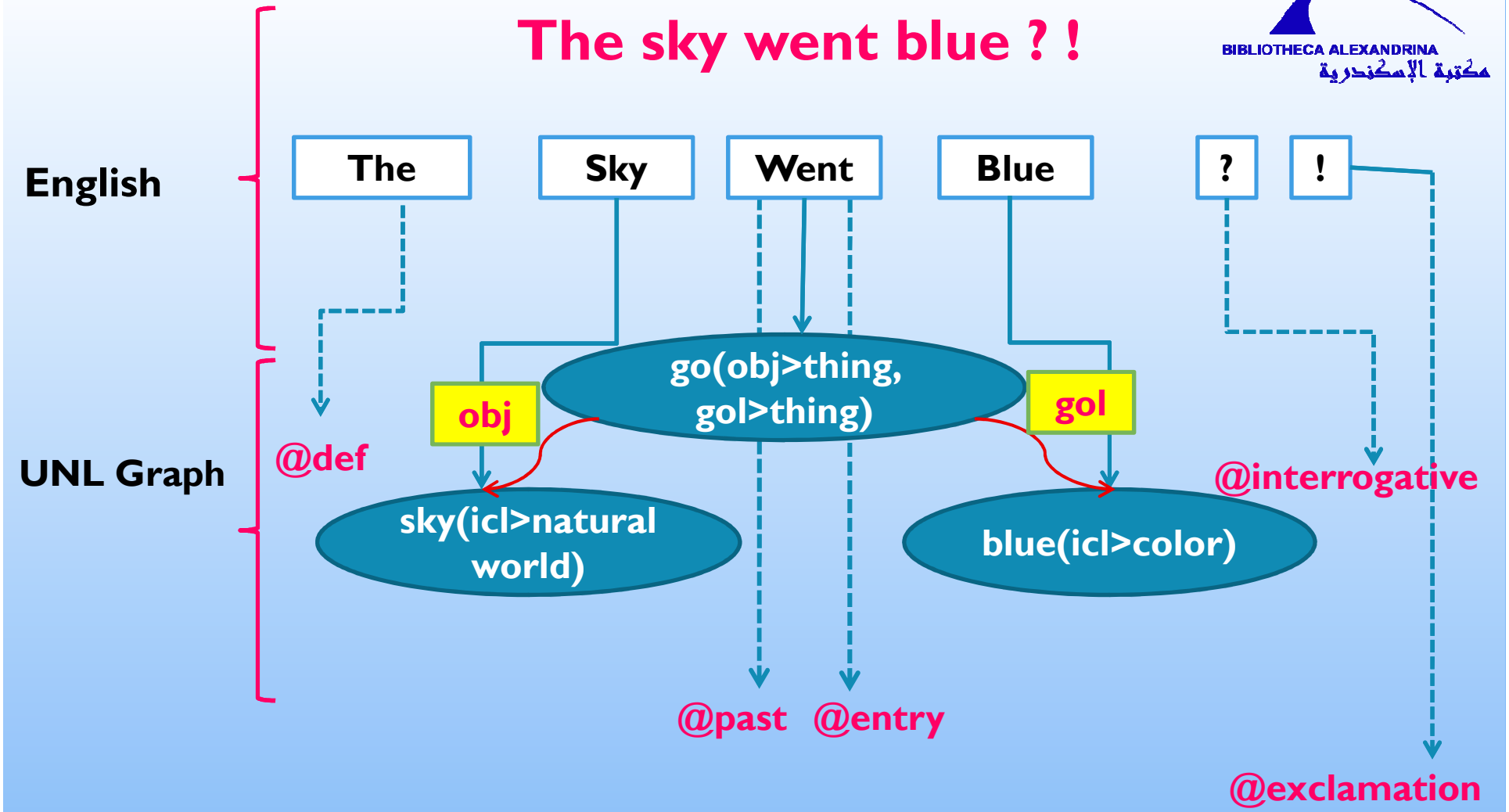
ما أغرب هذه الفكرة؟

4- Knowledge Base (UNLKB)



UW SYSTEM OF THE UNL KB

The sky went blue ? !



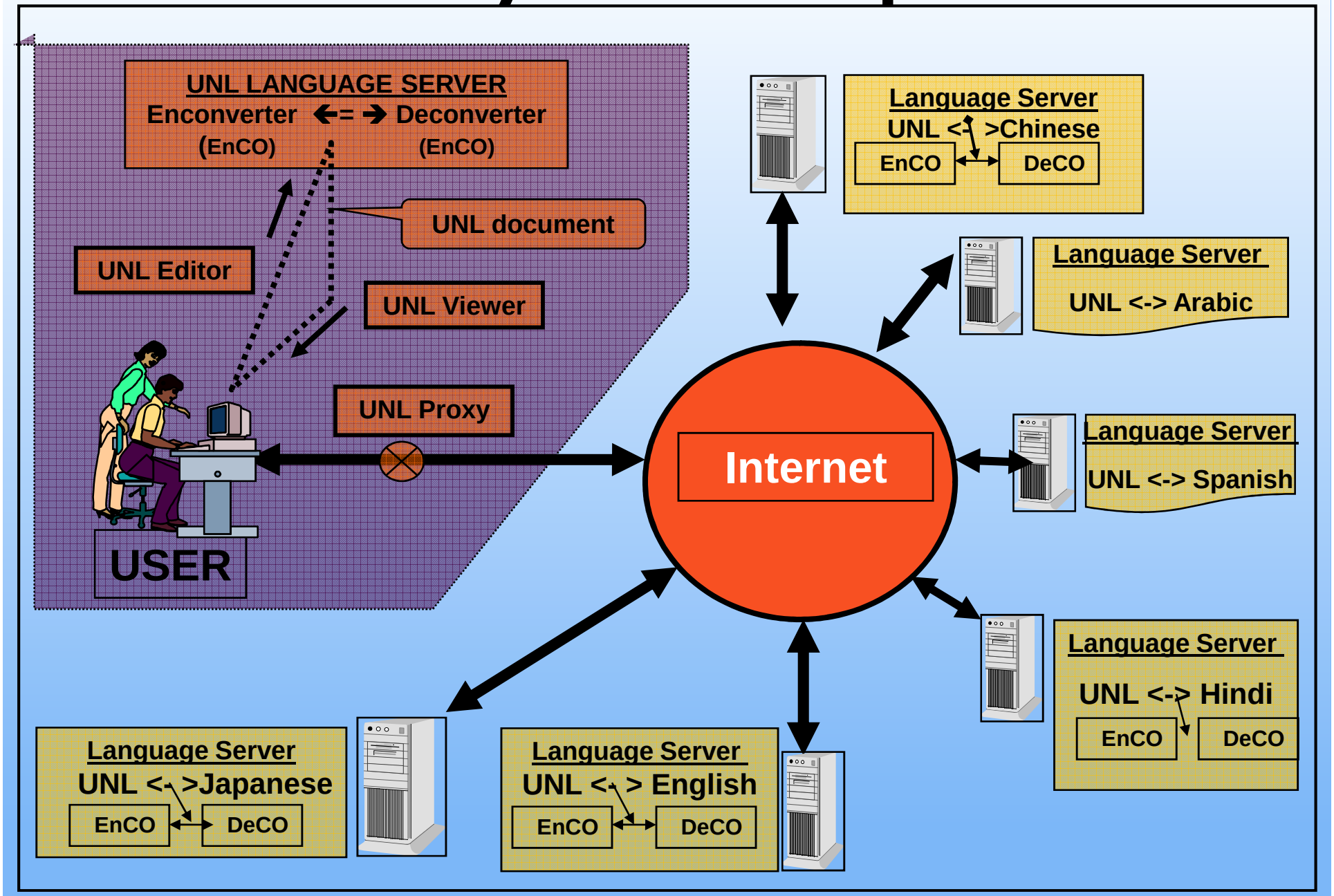
obj(go(obj>thing, gol>thing).@entry.@ interrogative.@exclamation:0K , sky(icl>natural world).@def:0P)
gol(go(obj>thing,gol>thing).@entry.@interrogative.@exclamation:0K,blue(icl>color):0X)

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

UNL as a System

- UNL Converters.
  - EnConverter.
  - DeConverter.
- Supporting Tools.

The UNL-system components

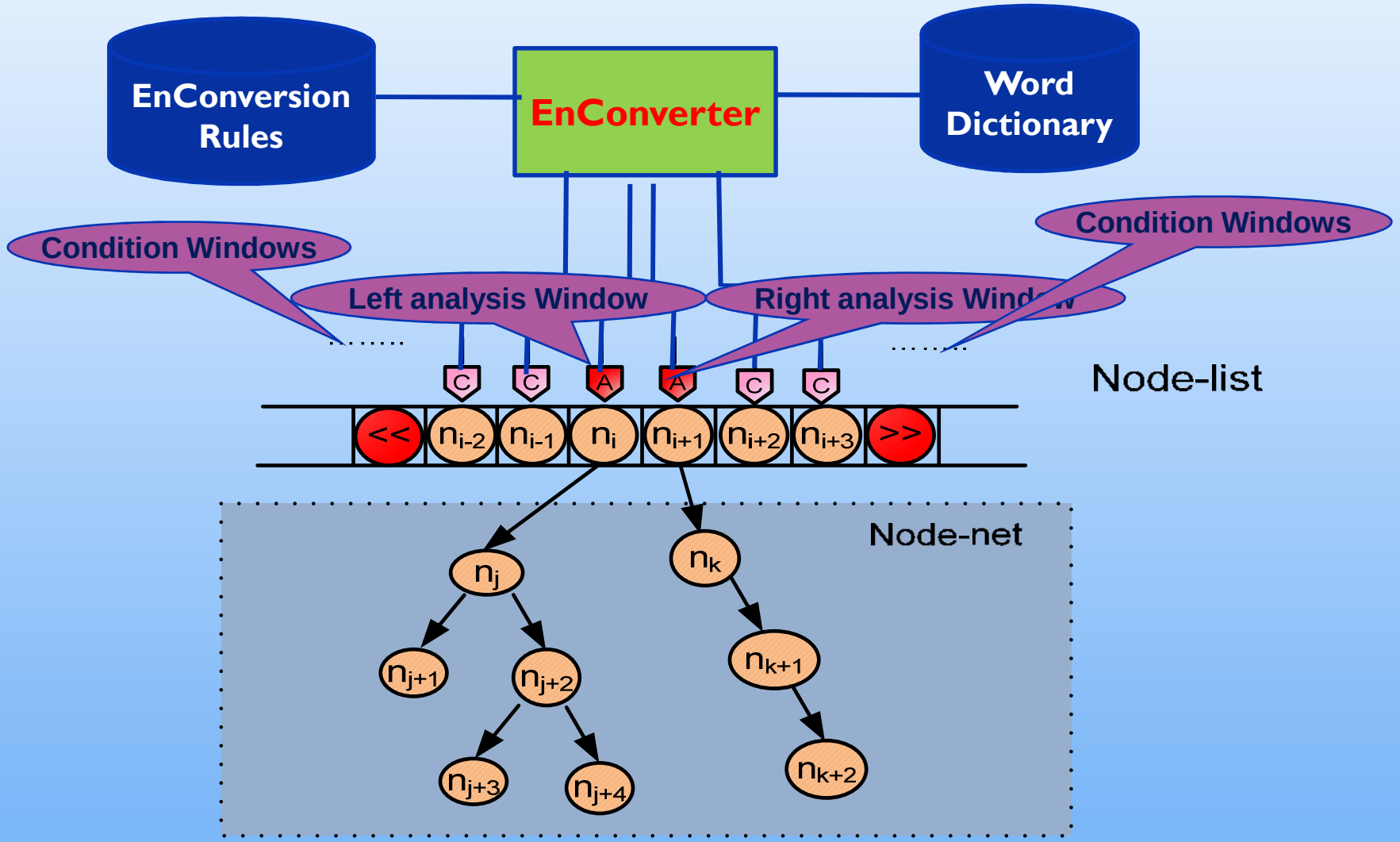


Enconverter

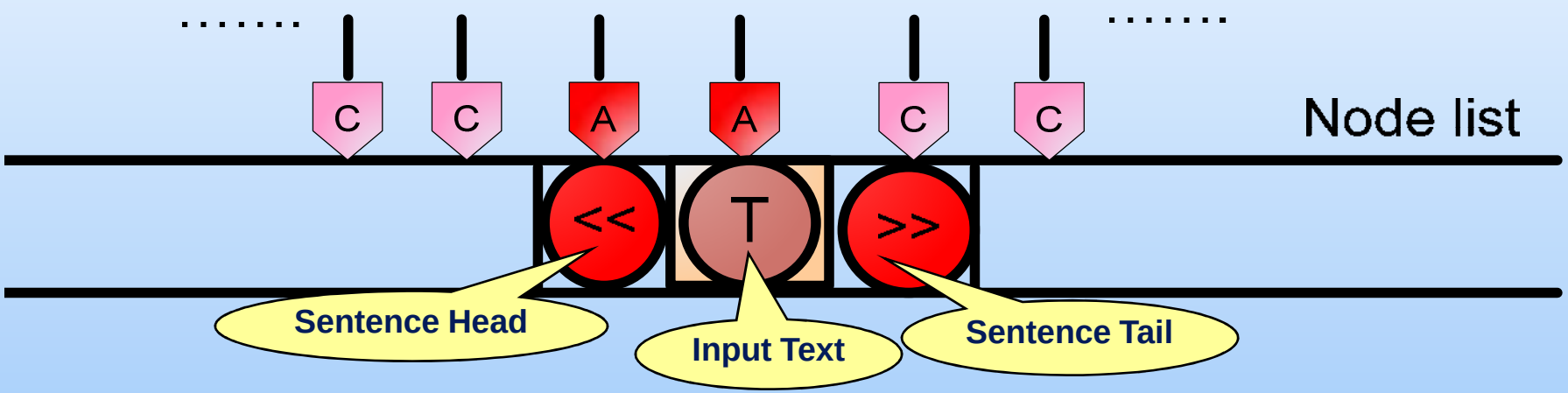
BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

- It is a language independent parser.
- It provides a framework for morphological and syntactic analysis.
- It is a software that automatically or interactively converts natural language texts into UNL expressions.

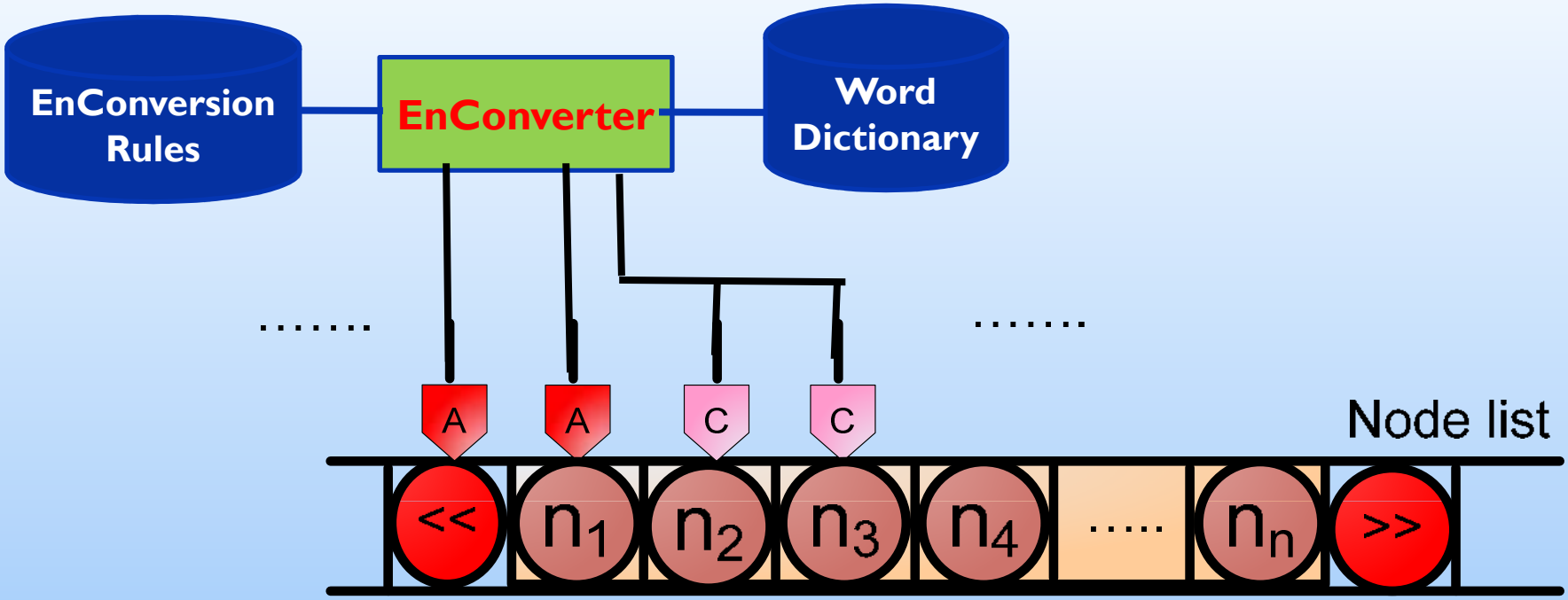
Structure of Enconverter:



How Enco engine works:

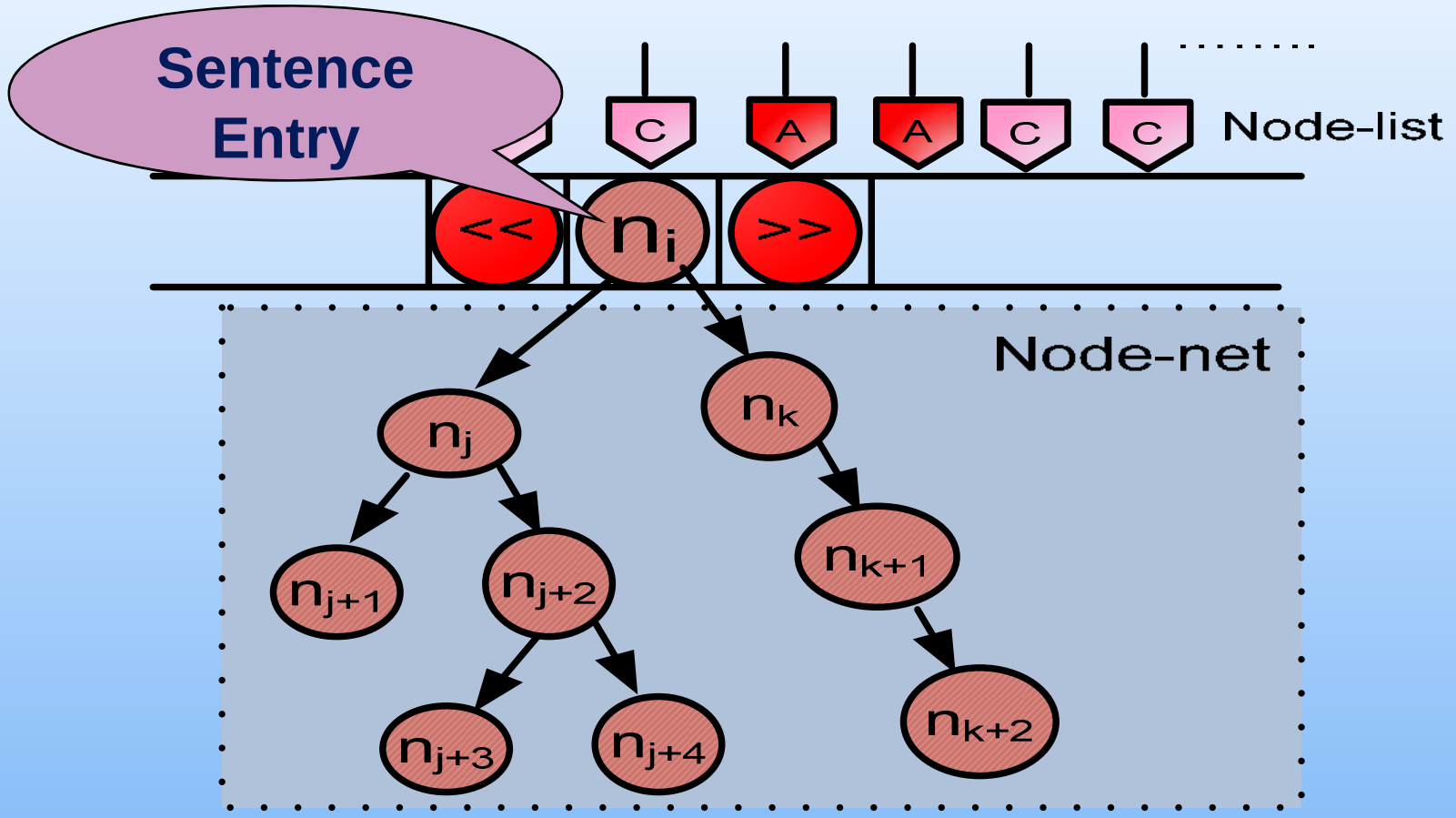


when a text of sentence is Input
(initial state)



when a list of morphemes of a sentence is input

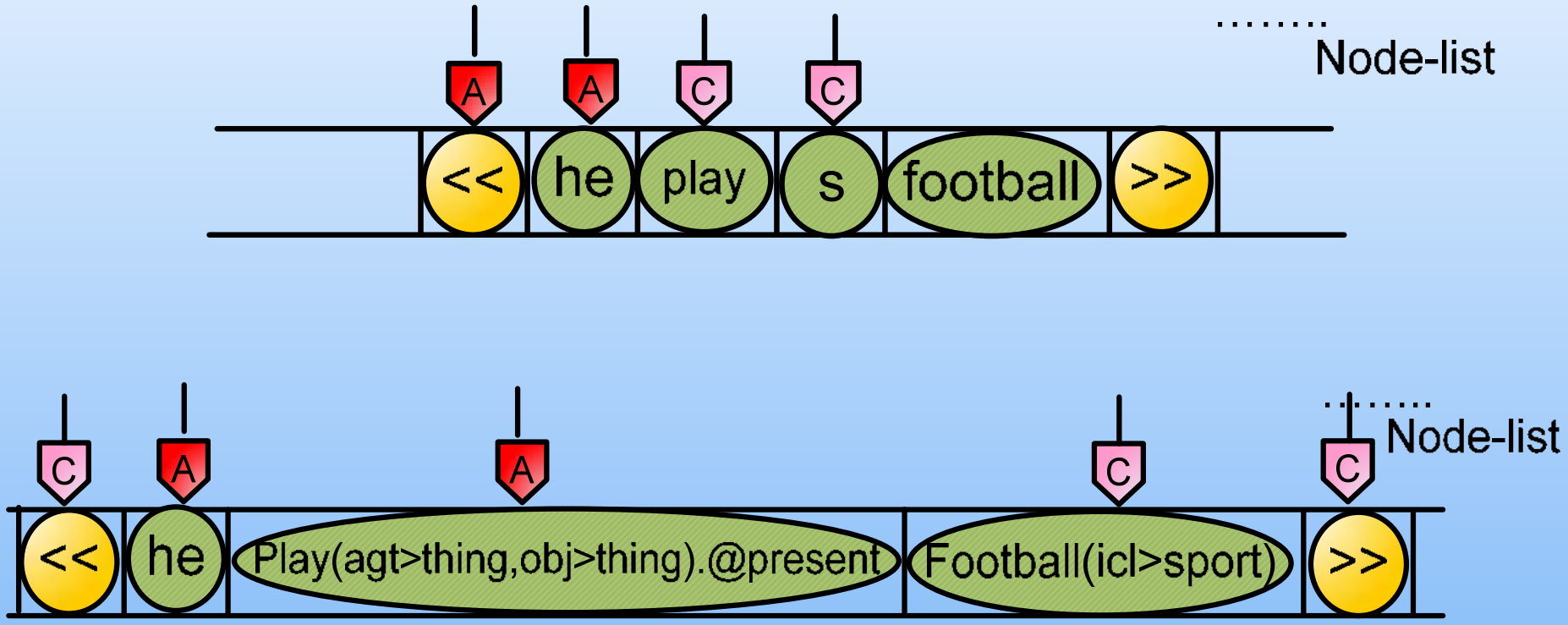
(Universal Word Extraction)



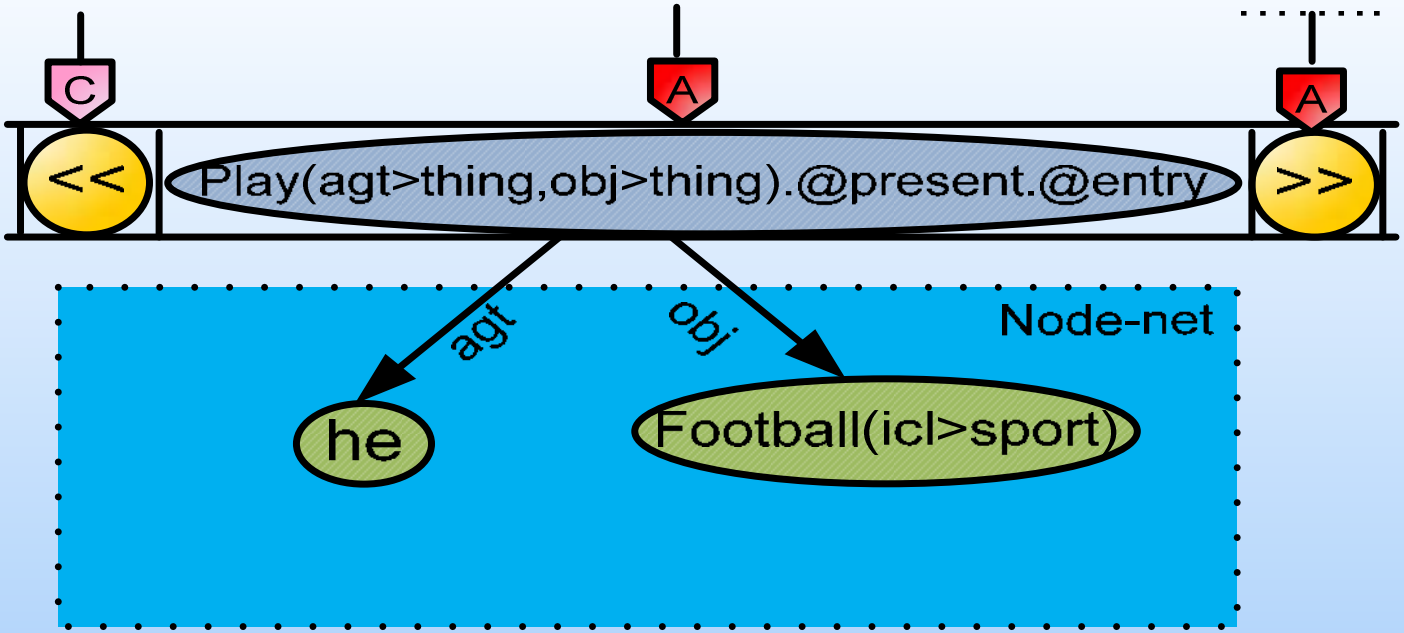
semantic network
(Final state)

Example:

He plays football.



(Universal word Extraction)



Building relations between the UWs

[s:1]
{org}
He plays football.
{/org}
{unl}
agt(play(agt>thing,obj>thing).@present,@entry:0U,he:0P)
obj(play(agt>thing,obj>thing).@present,@entry:0U,football(icl>sport):0W)
{/unl}
[/s]

Deconverter

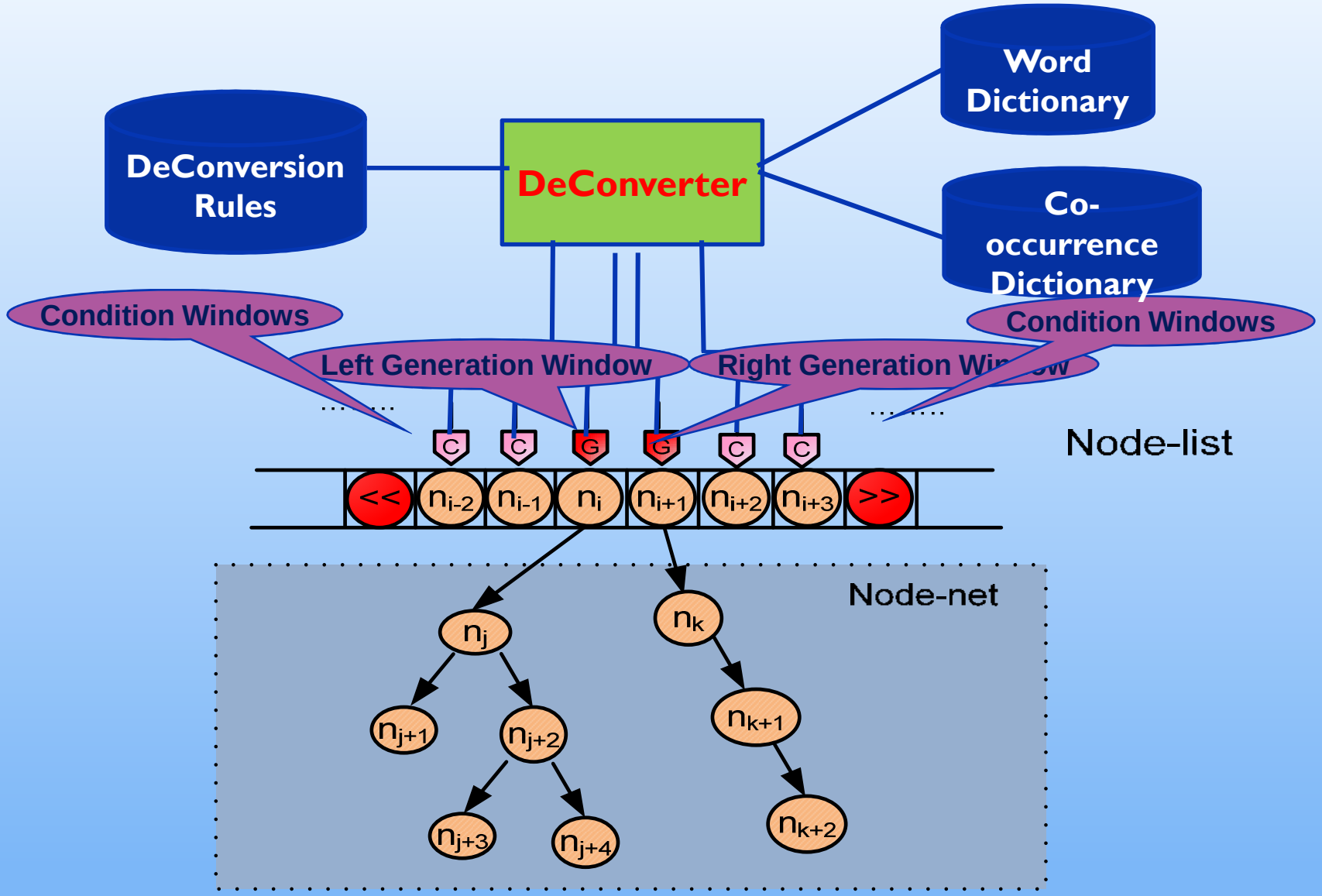
BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

- It is a language independent generator.

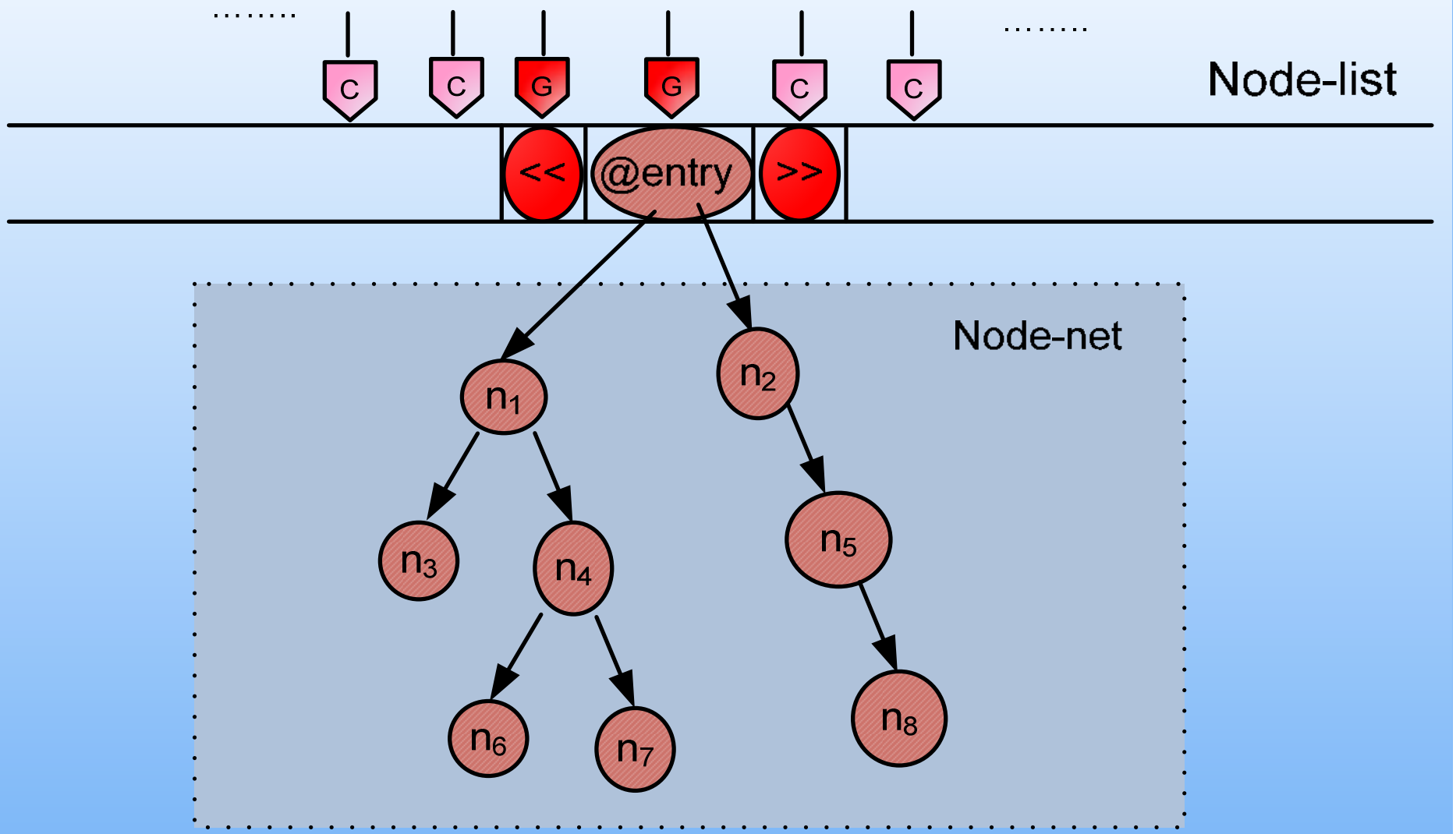
- It provides a framework for morphological and syntactic generation, and word selection for natural collection.

- It is a software that automatically deconverts UNL expressions into a variety of native languages using a different set of files .

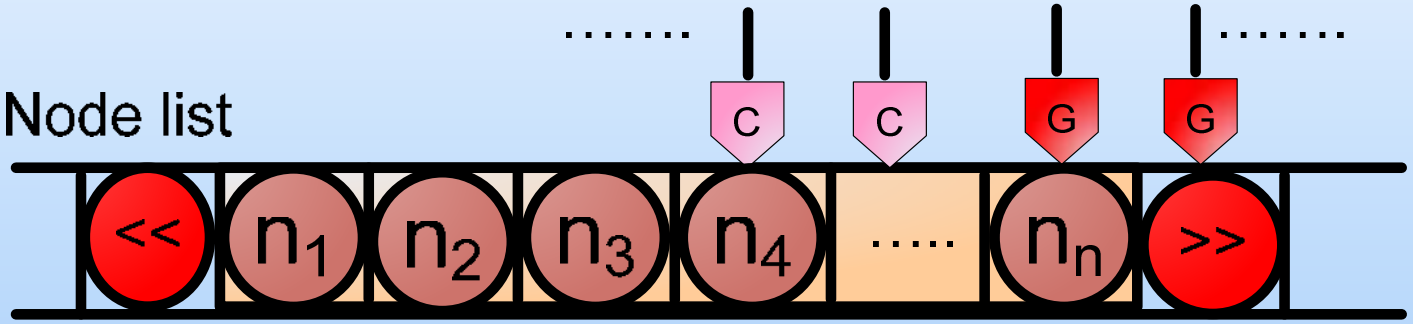
Structure of Deconverter:



How Deco engine works:



Initial state of Generation Windows and Node-List

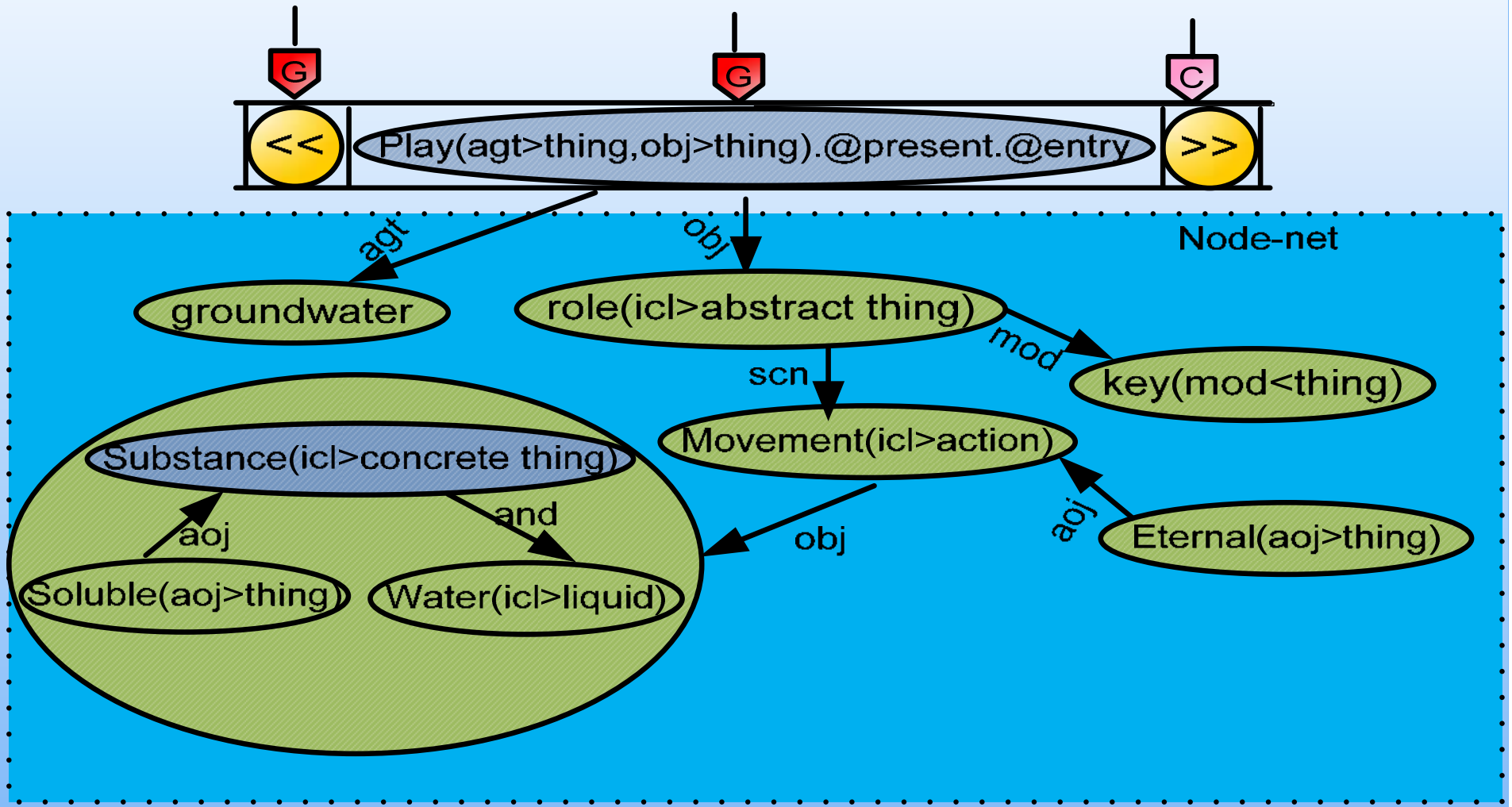


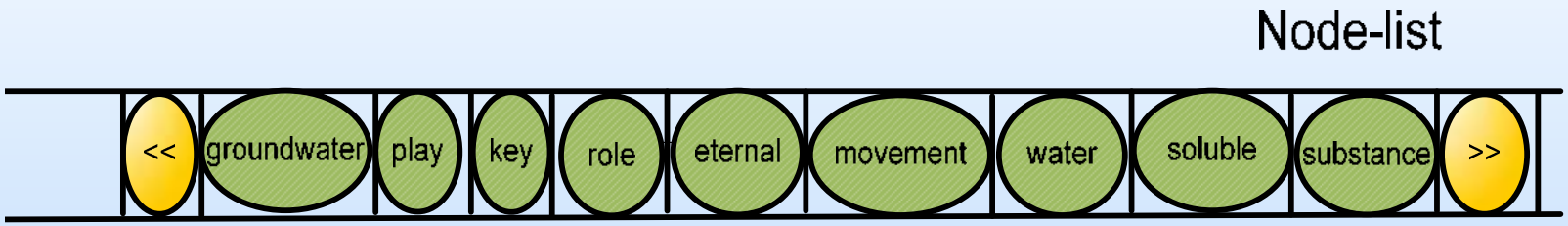
Final state of Generation Windows and Node-List

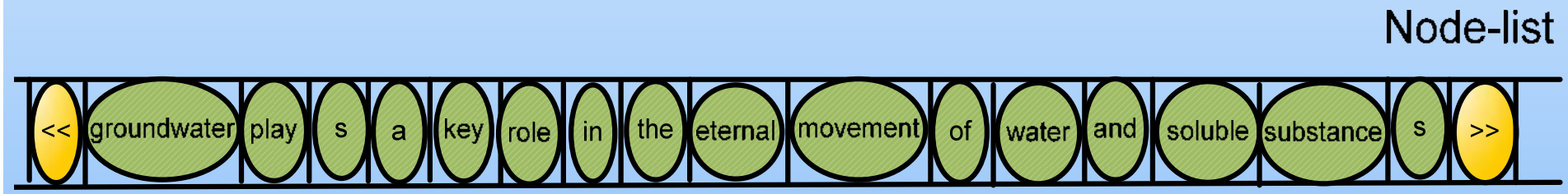
Example:

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

```
[s:1]
{unl}
agt(play(agt>thing,obj>thing):0E.@entry.@present,          groundwater:02)
obj(play(agt>thing,obj>thing):0E.@entry .@present,         role(icl>abstract thing):0Q.@indef)
mod(role(icl>abstract thing):0Q.@indef,  key(mod<thing):0M)
scn(role(icl>abstract thing):0Q.@indef, movement(icl>action):1B)
obj(movement(icl>action):1B, :01)
aoj(eternal(aoj>thing):13, movement(icl>action):1B)
and:01(substance(icl>concrete thing):25.@entry.@pl, water(icl>liquid):1N)
aoj:01(soluble(aoj>thing):1X, substance(icl>concrete thing):25.@entry.@pl)
{/unl}
[/s]
```







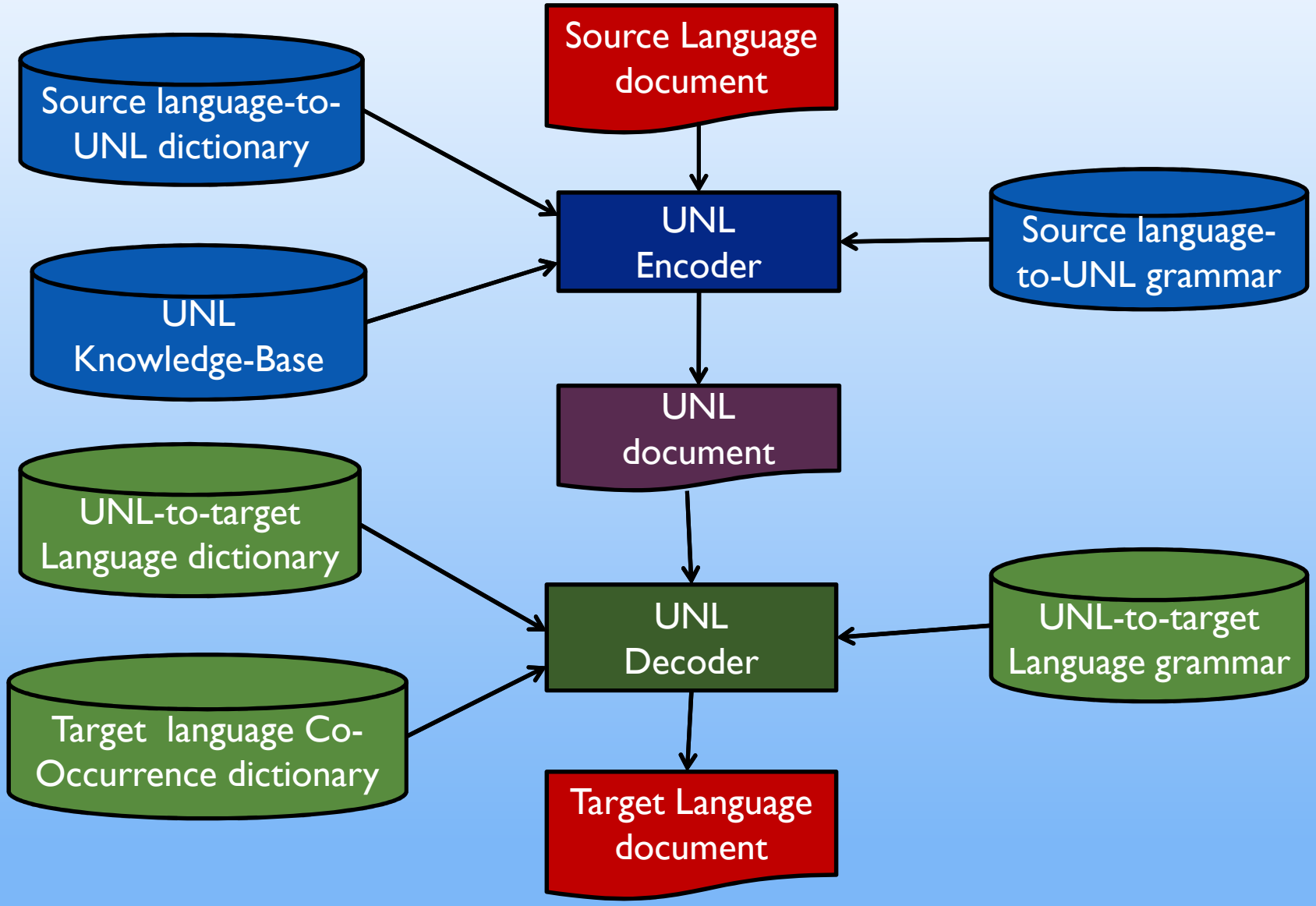
Groundwater plays a key role in the eternal movement of water and soluble substances.

BIBLIOTHECA ALEXANDRINA
مكتبة الإسكندرية

Supporting Tools

- UNL Language Server.
- UNL Proxy Server.
- UNL Editor.
- UNL Encyclopedia.
- UNL Verifier.
- UNL Explorer.
- UW Gate.
- UNL Viewer.

UNL System Architecture



Thank you

Introduction to language modeling

Dr. Mohamed Waleed Fakhr

AAST

Language Engineering Conference

22 December 2009

Topics

- Why a language model?
- Probability in brief
- Word prediction task
- Language modeling (N-grams)
  - N-gram intro.
  - Model evaluation
  - Smoothing
- Other modeling approaches

Why a language model?

- Suppose a machine is required to translate: “The human Race”.
- The word “Race” has at least 2 meanings, which one to choose?
- Obviously, the choice depends on the “history” or the “context” preceding the word “Race”. E.g., “the human race” versus “the dogs race”.
- A statistical language model can solve this ambiguity by giving higher probability to the correct meaning.

Probability in brief

- Joint probability: P(A,B) is the probability that events A and B are simultaneously true (observed together).
- Conditional probability: P(A|B): is the probability that A is true given that B is true (observed).

Relation between joint and conditional probabilities

- **BAYES RULE:**

$P(A|B) = P(A,B)/P(B)$

$P(B|A) = P(A,B)/P(A)$

Or;

$P(A,B) = P(A).P(B|A) = P(B).P(A|B)$

Chain Rule

- The joint probability:
  $P(A,B,C,D)=P(A).P(B|A).P(C|A,B).P(D|A,B,C)$
- This will lend itself to the language modeling paradigm as we will be concerned by the joint probability of the occurrence of a word-sequence $(W_1,W_2,W_3,\ldots.W_n)$:

  $P(W_1,W_2,W_3,\ldots.W_n)$

  which will be put in terms of conditional probability terms:
- P(W1).P(W2|W1).P(W3|W1,W2)……

  (More of this later)

Language Modeling?

In the narrow sense, statistical language modeling is concerned by estimating the joint probability of a word sequence . $P(W_1,W_2,W_3,\ldots W_n)$

This is always converted into conditional probs: P(Next Word | History)

e.g., P(W3|W1,W2)

i.e., can we predict the next word given the previous words that have been observed?

In other words, if we have a History, find the Next-Word that gives the highest prob.

Word Prediction

- Guess the next word...

  … It is too late I want to go ???

  ... I notice three guys standing on the ???

- There are many sources of knowledge that can be used to inform this task, including arbitrary world knowledge and deeper history (It is too late)
- But it turns out that we can do pretty well by simply looking at the preceding words and keeping track of some fairly simple counts.

Word Prediction

- We can formalize this task using what are called *N*-gram models.
- *N*-grams are token sequences of length *N*.
- Our 2nd example contains the following 2-grams (Bigrams)
  - (I notice), (notice three), (three guys), (guys standing), (standing on), (on the)
- Given knowledge of counts of N-grams such as these, we can guess likely next words in a sequence.

N-Gram Models

- More formally, we can use knowledge of the counts of *N*-grams to assess the conditional probability of candidate words as the next word in a sequence.
- In doing so, we actually use them to assess the joint probability of an entire sequence of words. (chain rule).

Applications

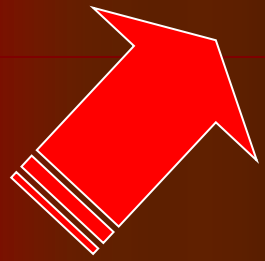
- It turns out that being able to predict the next word (or any linguistic unit) in a sequence is an extremely useful thing to be able to do.
- As we'll see, it lies at the core of the following applications
  - Automatic speech recognition
  - Handwriting and character recognition
  - Spelling correction
  - Machine translation
  - Information retrieval
  - And many more.

ASR

$$\arg\max_{wordsequence} P(wordsequence \mid acoustics) =$$

$$\arg\max_{wordsequence} \frac{P(acoustics \mid wordsequence) \times P(wordsequence)}{P(acoustics)}$$

$$\arg\max_{wordsequence} P(acoustics \mid wordsequence) \times P(wordsequence)$$



Source Channel Model for Machine Translation

$$\arg\max_{wordsequence} P(wordsequence \mid acoustics) =$$

$$\arg\max_{wordsequence} \frac{P(acoustics \mid wordsequence)' \; P(wordsequence)}{P(acoustics)}$$

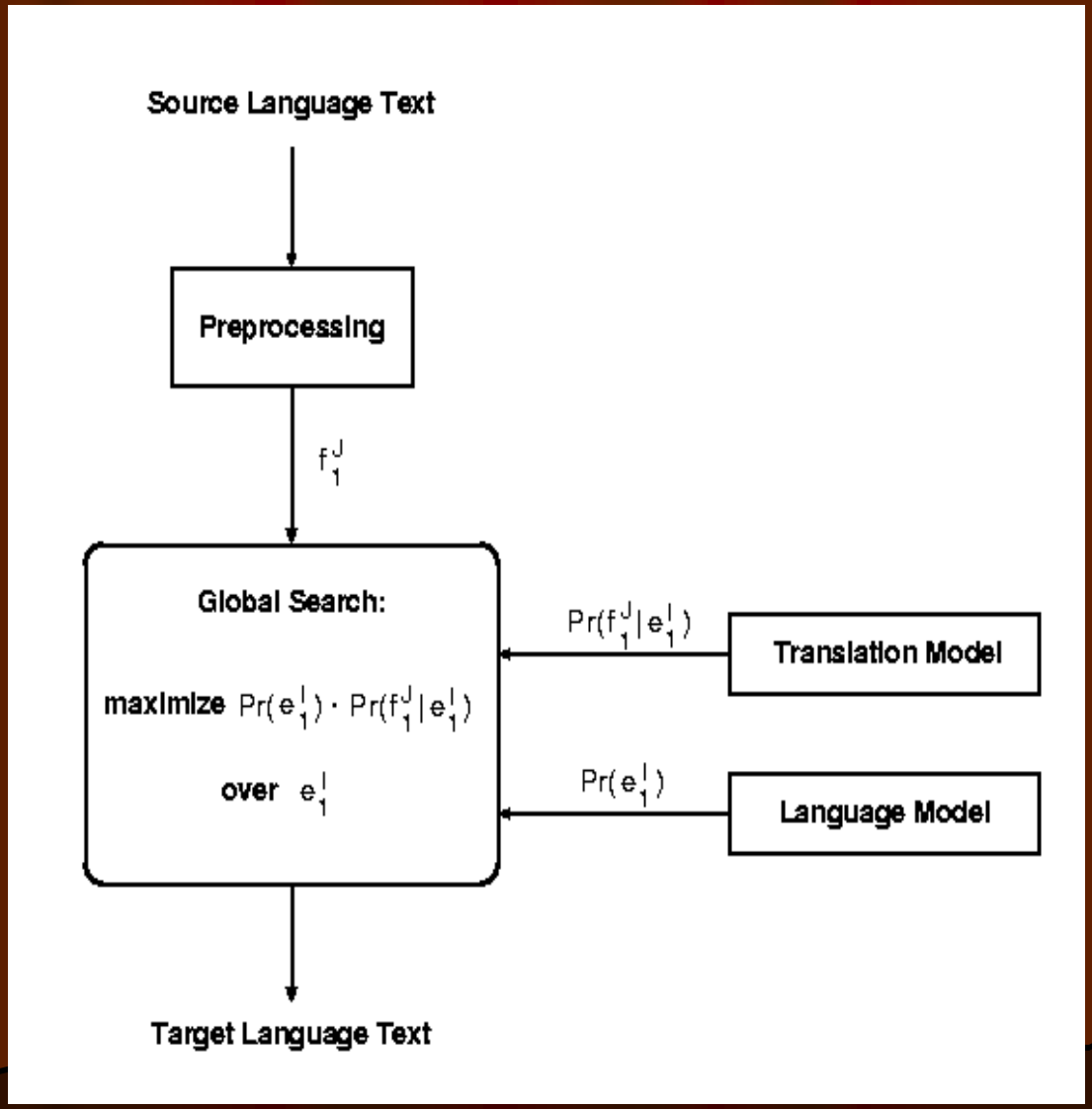
$$\arg\max_{wordsequence} P(acoustics \mid wordsequence)' \; P(wordsequence)$$

$$\arg\max_{wordsequence} P(english \mid french) =$$

$$\arg\max_{wordsequence} \frac{P(french \mid english)' \; P(english)}{P(french)}$$

$$\arg\max_{wordsequence} P(french \mid english)' \; P(english)$$

SMT Architecture



Based on Bayes´ Decision Rule:

$\hat{e} = \text{argmax}\{ p(e \mid f) \}$
$= \text{argmax}\{ p(e)\, p(f \mid e) \}$

Counting

- Simple counting lies at the core of any probabilistic approach. So let's first take a look at what we're counting.
  - *He stepped out into the hall, was delighted to encounter a water brother.*
    - 13 tokens, 15 if we include "," and "." as separate tokens.
    - Assuming we include the comma and period, how many bigrams are there?

Counting

- Not always that simple
  - *I do uh main- mainly business data processing*
- Spoken language poses various challenges.
  - Should we count “uh” and other fillers as tokens?
  - What about the repetition of “mainly”? Should such do-overs count twice or just once?
  - The answers depend on the application.
    - If we’re focusing on something like ASR to support indexing for search, then “uh” isn’t helpful (it’s not likely to occur as a query).
    - But filled pauses are very useful in dialog management, so we might want them there.

Counting: Types and Tokens

- How about
  - *They picnicked by the pool, then lay back on the grass and looked at the stars.*
    - 18 tokens (again counting punctuation)
- But we might also note that "*the*" is used 3 times, so there are only 16 unique types (as opposed to tokens).
- In going forward, we'll have occasion to focus on counting both types and tokens of both words and *N*-grams.

Counting: Wordforms

- Should “cats” and “cat” count as the same when we’re counting?
- How about “geese” and “goose”?
- Some terminology:
  - Lemma: a set of lexical forms having the same stem, major part of speech, and rough word sense: (car, cars, automobile)
  - Wordform: fully inflected surface form
- Again, we’ll have occasion to count both lemmas, morphemes, and wordforms

Counting: Corpora

- So what happens when we look at large bodies of text instead of single utterances?
- Brown et al (1992) large corpus of English text
  - 583 million wordform tokens
  - 293,181 wordform types
- Google
  - Crawl of 1,024,908,267,229 English tokens
  - 13,588,391 wordform types
    - That seems like a lot of types… After all, even large dictionaries of English have only around 500… …here?

- Numbers
- Misspellings
- Names
- Acronyms
- etc

Language Modeling

- Back to word prediction
- We can model the word prediction task as the ability to assess the conditional probability of a word given the previous words in the sequence
  - $P(w_n|w_1,w_2 \ldots w_{n-1})$
- We'll call a statistical model that can assess this a *Language Model*

Language Modeling

- How might we go about calculating such a conditional probability?
  - One way is to use the definition of conditional probabilities and look for counts. So to get
  - P(*the* | *its water is so transparent that*)
- By definition that's

$$\frac{\text{Count(its water is so transparent that the)}}{\text{Count(its water is so transparent that)}}$$

We can get each of those counts in a large corpus.

Very Easy Estimate

- According to Google those counts are 5/9.
  - Unfortunately... 2 of those were to these slides... So maybe it's really 3/7
  - In any case, that's not terribly convincing due to the small numbers involved.

Language Modeling

- Unfortunately, for most sequences and for most text collections we won't get good estimates from this method.
  - What we're likely to get is 0. Or worse 0/0.
- Clearly, we'll have to be a little more clever.
  - Let's use the chain rule of probability
  - And a particularly useful independence assumption.

The Chain Rule

- Recall the definition of conditional probabilities
- Rewriting:

$$P(A \mid B) = \frac{P(A, B)}{P(B)}$$

$$P(A, B) = P(B).P(A \mid B)$$

- For sequences...
  - $P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$
- In general
  - $P(x_1,x_2,x_3,\ldots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1,x_2)\ldots P(x_n|x_1\ldots x_{n-1})$

The Chain Rule

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1})$$
$$= \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

P(its water was so transparent)=

P(its)*

P(water|its)*

P(was|its water)*

P(so|its water was)*

P(transparent|its water was so)

Unfortunately

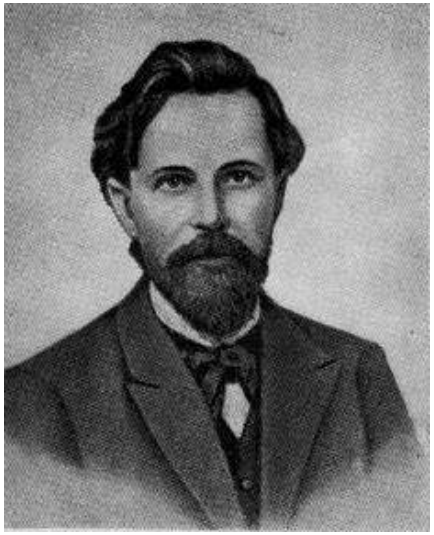
- There are still a lot of possible sentences
- In general, we'll never be able to get enough data to compute the statistics for those longer prefixes
  - Same problem we had for the strings themselves

Independence Assumption

- Make the simplifying assumption
  - P(lizard|the,other,day,I,was,walking,along,and ,saw,a) = P(lizard|a)
- Or maybe
  - P(lizard|the,other,day,I,was,walking,along,and ,saw,a) = P(lizard|saw,a)
- That is, the probability in question is independent of its earlier history.

Independence Assumption

- This particular kind of independence assumption is called a *Markov assumption* after the Russian mathematician Andrei Markov.



Markov Assumption

So for each component in the product replace with the approximation (assuming a prefix of N)

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

Bigram version

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-1})$$

Estimating Bigram Probabilities

- The Maximum Likelihood Estimate (MLE):

$$P(w_i \mid w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

Normalization

- For N-gram models to be probabilistically correct they have to obey prob. Normalization constraints:

$$\sum_{over-all-j} P(W_j \mid Context_i) = 1$$

- The sum over all words for the same context (history) must be 1.
- The context may be one word (bigram) or two words (trigram) or more.

An Example: bigrams

- <s> I am Sam </s>
- <s> Sam I am </s>
- <s> I do not like green eggs and ham </s>

$$P(\mathtt{I}\,|\,\mathtt{<s>}) = \tfrac{2}{3} = .67 \qquad P(\mathtt{Sam}\,|\,\mathtt{<s>}) = \tfrac{1}{3} = .33 \qquad P(\mathtt{am}\,|\,\mathtt{I}) = \tfrac{2}{3} = .67$$
$$P(\mathtt{</s>}\,|\,\mathtt{Sam}) = \tfrac{1}{2} = 0.5 \qquad P(\mathtt{Sam}\,|\,\mathtt{am}) = \tfrac{1}{2} = .5 \qquad P(\mathtt{do}\,|\,\mathtt{I}) = \tfrac{1}{3} = .33$$

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

estimates depend on the corpus

- The maximum likelihood estimate of some parameter of a model M from a training set T
  - Is the estimate that maximizes the likelihood of the training set T given the model M
- Suppose the word Chinese occurs 400 times in a corpus of a million words (Brown corpus)
- What is the probability that a random word from some other text from the same distribution will be “Chinese”
- MLE estimate is 400/1000000 = .004
  - This may be a bad estimate for some other corpus

Berkeley Restaurant Project Sentences examples

- *can you tell me about any good cantonese restaurants close by*
- *mid priced thai food is what i'm looking for*
- *tell me about chez panisse*
- *can you give me a listing of the kinds of food that are available*
- *i'm looking for a good place to eat breakfast*
- *when is caffe venezia open during the day*

Bigram Counts

- Out of 9222 sentences
  - e.g. “I want” occurred 827 times

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Bigram Probabilities

- Divide bigram counts by prefix unigram counts to get probabilities.

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

examples

- P(Want | I) = C(I Want) / C(I)

 = 827/2533 = 0.33

P(Food | Chinese) = C(Chinese Food) / C(Chinese)

= 82/158 = 0.52

Bigram Estimates of Sentence Probabilities

- P(<s> I want english food </s>) =
  P(i|<s>)*
  P(want|I)*
  P(english|want)*
  P(food|english)*
  P(</s>|food)*
  =.000031

Evaluation

- How do we know if our models are any good?
  - And in particular, how do we know if one model is better than another?

Evaluation

- Standard method
  - Train parameters of our model on a **training set**.
  - Look at the models performance on some new data
    - This is exactly what happens in the real world; we want to know how our model performs on data we haven't seen
  - So use a **test set**. A dataset which is different than our training set, but is drawn from the same source
  - Then we need an **evaluation metric** to tell us how well our model is doing on the test set.
    - One such metric is **perplexity**

Unknown Words

- But once we start looking at test data, we'll run into words that we haven't seen before (pretty much regardless of how much training data you have) (zero unigrams)
- With an ***Open Vocabulary* task**
  - Create an unknown word token <UNK>
  - Training of <UNK> probabilities
    - Create a fixed lexicon L, of size V
      - From a dictionary or
      - A subset of terms from the training set
    - At text normalization phase, any training word not in L changed to **<UNK>**
    - Now we count that like a normal word
  - At test time
    - Use <**UNK>** counts for any word not in training

Perplexity

- Perplexity is the probability of the test set (assigned by the language model), normalized by the number of words:

$$\begin{aligned} \mathrm{PP}(W) &= P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}} \end{aligned}$$

- Chain rule:

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_1 \ldots w_{i-1})}}$$

- For bigrams:

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1})}}$$

- Minimizing perplexity is the same as maximizing probability
  - **The best language model is one that best predicts an unseen test set**

Lower perplexity means a better model

- Training 38 million words, test 1.5 million words, WSJ (Wall-Street Journal)

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

Evaluating *N*-Gram Models

- Best evaluation for a language model
  - Put model *A* into an application
    - For example, a speech recognizer
  - Evaluate the performance of the application with model *A*
  - Put model *B* into the application and evaluate
  - Compare performance of the application with the two models
  - ***Extrinsic evaluation***

Difficulty of extrinsic (in-vivo) evaluation of N-gram models

- Extrinsic evaluation
  - This is really time-consuming
  - Can take days to run an experiment
- So
  - To evaluate N-grams we often use an **intrinsic** evaluation, an approximation called **perplexity**
  - But perplexity is a poor approximation unless the test data looks **similar to** the training data
  - So is **generally only useful in pilot experiments**
  - **But still, there is nothing like the real experiment!**

N-gram Zero Counts

- For the English language,
  - V^2= 844 million possible bigrams...
  - So, for a medium size training data, e.g., Shakespeare novels, 300,000 bigrams were found Thus, 99.96% of the possible bigrams were never seen (have zero entries in the table)
  - Does that mean that any ***test*** sentence that contains one of those bigrams should have a probability of 0?

N-gram Zero Counts

- Some of those zeros are really zeros...
  - Things that really can't or shouldn't happen.
- On the other hand, some of them are just rare events.
  - If the training corpus had been a little bigger they would have had a count (probably a count of 1).
- Zipf's Law (long tail phenomenon):
  - A small number of events occur with high frequency
  - A large number of events occur with low frequency
  - You can quickly collect statistics on the high frequency events
  - You might have to wait an arbitrarily long time to get valid statistics on low frequency events
- Result:
  - Our estimates are sparse ! We have no counts at all for the vast bulk of things we want to estimate!
- Answer:
  - ***Estimate*** the likelihood of unseen (zero count) N-grams!
  - **N-gram Smoothing techniques**

Laplace Smoothing

- Also called add-one smoothing
- Just add one to all the counts!
- This adds extra *V* observations (*V* is vocab. Size)

- MLE estimate: $$P(w_i) = \frac{c_i}{N}$$

- Laplace estimate: $$P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$ $$P_{Laplace} = \frac{1}{N}\frac{(ci+1).N}{(N+V)}$$

- Reconstructed counts:

(making the volume N again) $$c_i^* = (c_i + 1)\frac{N}{N + V}$$

Laplace-Smoothed Bigram Counts

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Laplace-Smoothed Bigram Probabilities

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Reconstructed Counts

$$c^*(w_{n-1}w_n) = \frac{[C(w_{n-1}w_n)+1] \times C(w_{n-1})}{C(w_{n-1})+V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

$$P(w1 \mid w2) = \frac{C(w2w1)+1}{C(w2)+V} = \frac{C(w2)}{C(w2)} \frac{C(w2w1)+1}{C(w2)+V} = \frac{1}{C(w2)} \frac{C(w2).[C(w2w1)+1]}{[C(w2)+V]}$$

Big Change to the Counts!

- C(want to) went from 608 to 238!
- P(to|want) from .66 to .26!
- Discount d= c*/c
  - d for “Chinese food” = 0.1 !!! A 10x reduction
  - So in general, Laplace is a blunt instrument
  - Could use more fine-grained method (add-k)
- But Laplace smoothing not used for N-grams, as we have much better methods
- Despite its flaws, Laplace (add-k) is however still used to smooth other probabilistic models in NLP, especially
  - For pilot studies
  - in domains where the number of zeros isn’t so huge.

Better Smoothing

- Intuition used by many smoothing algorithms, for example;
  - Good-Turing
  - Kneyser-Ney
  - Witten-Bell
- Is to use the count of things we've seen ***once*** to help estimate the count of things we've never seen

Good-Turing
Josh Goodman Intuition

- Imagine you are fishing
  - There are 8 species in this waters: carp, perch, whitefish, trout, salmon, eel, catfish, bass
- You have caught
  - 10 carp, 3 perch, 2 whitefish, 1 trout, 1 salmon, 1 eel = 18 fish
- How likely is it that the next fish caught is from a new species (one not seen in our previous catch)?
  - 3/18 (3 is number of events that seen once)
- Assuming so, how likely is it that next species is trout?
  - Must be less than 1/18 because we just stole 3/18 of our probability mass to use on unseen events

Good-Turing

Notation: Nx is the frequency-of-frequency-x

So N**10**=1

Number of fish species seen 10 times is 1 (carp)

N**1**=3

Number of fish species seen 1 time is 3 (trout, salmon, eel)

To estimate total number of unseen species (seen 0 times)

Use number of species (bigrams) we've seen once (i.e. 3)

So, the estimated count c* for <unseen> is 3.

All other estimates are adjusted (down) to account for the stolen mass given for the unseen events, using the formula:

$$c^* = (c+1)\frac{N_{c+1}}{N_c}$$

GT Fish Example

c	0	1	2
MLE p	0/18	1/18	2/18
c^*	$1 \times \frac{3}{1} = 3$	$2 \times \frac{1}{3} = .67$	$3 \times \frac{1}{1} = 3$
GT p^*	$\frac{3}{18} = .17$	$\frac{.67}{18} = .037$	$\frac{3}{18} = .17$

$$c^* = (c+1)\frac{N_{c+1}}{N_c}$$

Bigram Frequencies of Frequencies and GT Re-estimates

AP Newswire			Berkeley Restaurant—		
c (MLE)	N_c	c^* (GT)	c (MLE)	N_c	c^* (GT)
0	74,671,100,000	0.0000270	0	2,081,496	0.002553
1	2,018,046	0.446	1	5315	0.533960
2	449,721	1.26	2	1419	1.357294
3	188,933	2.24	3	642	2.373832
4	105,668	3.24	4	381	4.081365
5	68,379	4.22	5	311	3.781350
6	48,190	5.19	6	196	4.500000

AP Newswire: 22million words, Berkeley: 9332 sentences

Backoff and Interpolation

- Another really useful source of knowledge
- If we are estimating:
  - trigram p(z|x,y)
  - but count(xyz) is zero
- Use info from:
  - Bigram p(z|y)
- Or even:
  - Unigram p(z)
- How to combine this trigram, bigram, unigram info in a valid fashion?

Backoff Vs. Interpolation

1. **Backoff**: use trigram if you have it, otherwise bigram, otherwise unigram
2. **Interpolation**: mix all three by weights

Interpolation

- Simple interpolation

$$\begin{aligned}\hat{P}(w_n|w_{n-1}w_{n-2}) &= \lambda_1 P(w_n|w_{n-1}w_{n-2}) \\ &+\lambda_2 P(w_n|w_{n-1}) \\ &+\lambda_3 P(w_n)\end{aligned}$$

$$\sum_i \lambda_i = 1$$

- Lambdas conditional on context:

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) &= \lambda_1(w_{n-2}^{n-1}) P(w_n|w_{n-2}w_{n-1}) \\ &+\lambda_2(w_{n-2}^{n-1}) P(w_n|w_{n-1}) \\ &+\lambda_3(w_{n-2}^{n-1}) P(w_n)\end{aligned}$$

How to Set the Lambdas?

- Use a **held-out, or development** corpus
- Choose lambdas which maximize the probability of some held-out data
  - I.e. fix the *N*-gram probabilities
  - Then search for lambda values that when plugged into previous equation give largest probability for held-out set
  - Can use EM to do this search
  - Can use direct search methods (Genetic, Swarm, etc…)

Katz Backoff (very popular)

$$P_{\text{katz}}(w_n|w_{n-N+1}^{n-1}) = \begin{cases} P^*(w_n|w_{n-N+1}^{n-1}), & \text{if } C(w_{n-N+1}^{n}) > 0 \\ \alpha(w_{n-N+1}^{n-1}) P_{\text{katz}}(w_n|w_{n-N+2}^{n-1}), & \text{otherwise.} \end{cases}$$

$$P_{\text{katz}}(z|x,y) = \begin{cases} P^*(z|x,y), & \text{if } C(x,y,z) > 0 \\ \alpha(x,y) P_{\text{katz}}(z|y), & \text{else if } C(x,y) > 0 \\ P^*(z), & \text{otherwise.} \end{cases}$$

$$P_{\text{katz}}(z|y) = \begin{cases} P^*(z|y), & \text{if } C(y,z) > 0 \\ \alpha(y) P^*(z), & \text{otherwise.} \end{cases}$$

Why discounts P* and alpha?

- MLE probabilities sum to 1

$$\sum_i P(w_i|w_j w_k) = 1$$

- So if we used MLE probabilities but backed off to lower order model when MLE prob is zero we would be adding extra probability mass (it is like in smoothing), and total probability would be greater than 1. So, we have to do discounting.

OOV words: <UNK> word

- **Out Of Vocabulary** = OOV words
- create an unknown word token <UNK>
  - Training of <UNK> probabilities
    - Create a fixed lexicon L of size V
    - At text normalization phase, any training word not in L changed to <UNK>
    - Now we train its probabilities like a normal word
  - At decoding time
    - If text input: Use UNK probabilities for any word not in training

Other Approaches

Class-based LMs
Morpheme-based LMs
Skip LMs

Class-based Language Models

- Standard word-based language models

$$p(w_1, w_2, ..., w_T) = \prod_{t=1}^{T} p(w_t \mid w_1, ..., w_{t-1})$$

$$\approx \prod_{t=1}^{T} p(w_t \mid w_{t-1}, w_{t-2})$$

- How to get robust n-gram estimates ($p(w_t \mid w_{t-1}, w_{t-2})$)?
  - Smoothing
    - E.g. Kneyser-Ney, Good-Turing
  - Class-based language models

$$p(w_t \mid w_{t-1}) \approx p(w_t \mid C(w_t)) p(C(w_t) \mid C(w_{t-1}))$$

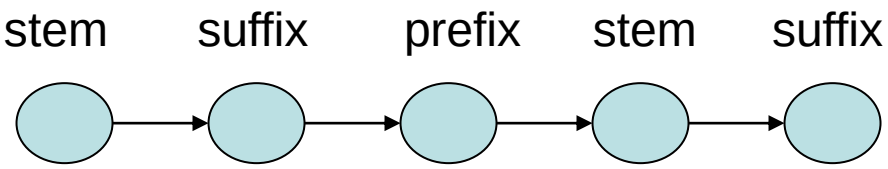
Limitation of Word-based Language Models

- **Words are inseparable whole units**.
  - E.g. “book” and “books” are distinct vocabulary units
- Especially problematic in **morphologically-rich languages**:
  - E.g. Arabic, Finnish, Russian, Turkish
  - Many unseen word contexts
  - High out-of-vocabulary rate
  - High perplexity

Arabic k-t-b	
Kitaab	A book
Kitaab-iy	My book
Kitaabu-hum	Their book
Kutub	Books

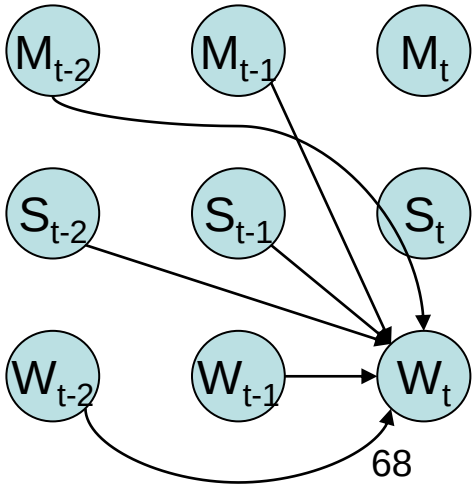
Solution: Word as Factors

- Decompose words into “factors” (e.g. stems)
- Build language model over factors: P(w|factors)
- Two approaches for decomposition
  - Linear
    - [e.g. Geutner, 1995]
  - Parallel



[Kirchhoff et. al., JHU Workshop 2002]

[Bilmes & Kirchhoff, NAACL/HLT 2003]



Different Kinds of Language Models

- cache language models (constantly adapting to a floating text)
- trigger language models (can handle long distance effects)
- POS-based language models, LM over POS tags
- class-based language models based on semantic classes
- multilevel *n*-gram language models (mix many LM together)
- interleaved language models (different LM for different parts of text)
- morpheme-based language models (separate words into core and modifyers)
- context free grammar language models (use simple and efficient LM-definition)
- decision tree language models (handle long distance effects, use rules)
- HMM language models (stochastic decision for combination of independent LMs)

OPTIMIZATION TECHNIQUES FOR SPEECH EMOTION RECOGNITION

Julia Sidorova

December 23, 2009

Speech Emotion Recognition

DEF.: Speech Emotion Recognition (SER): produce an estimate of the emotional state of the speaker given a speech fragment as input;
[physical changes $\longrightarrow$ measure $\longrightarrow$ feature vector $\longrightarrow$ pattern recognition]
MOTIVATION: HCI, robotics, smart call centres, etc.

ESEDA: speech emotion recognition system

INPUT

⇓

FEATURE EXTRACTION

- global statistics
- 116 features

⇓

FEATURE SELECTION

- correlation-based feature subset selection

⇓

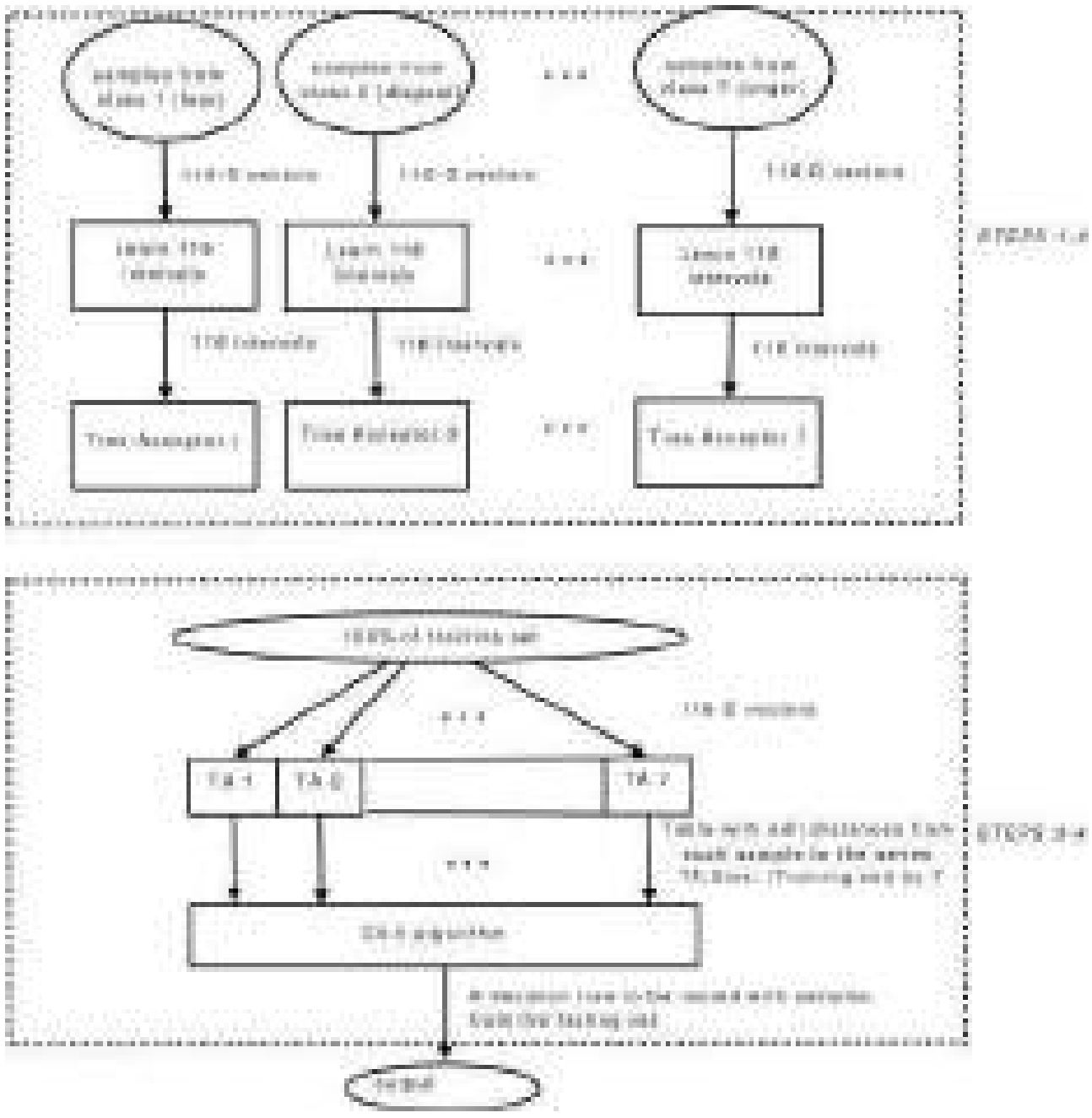
CLASSIFICATION

- weka's top performer: Multilayer Perceptron or Support Vector Machine

⇓

OUTPUT

Idea from OCR: tree automata + decision trees

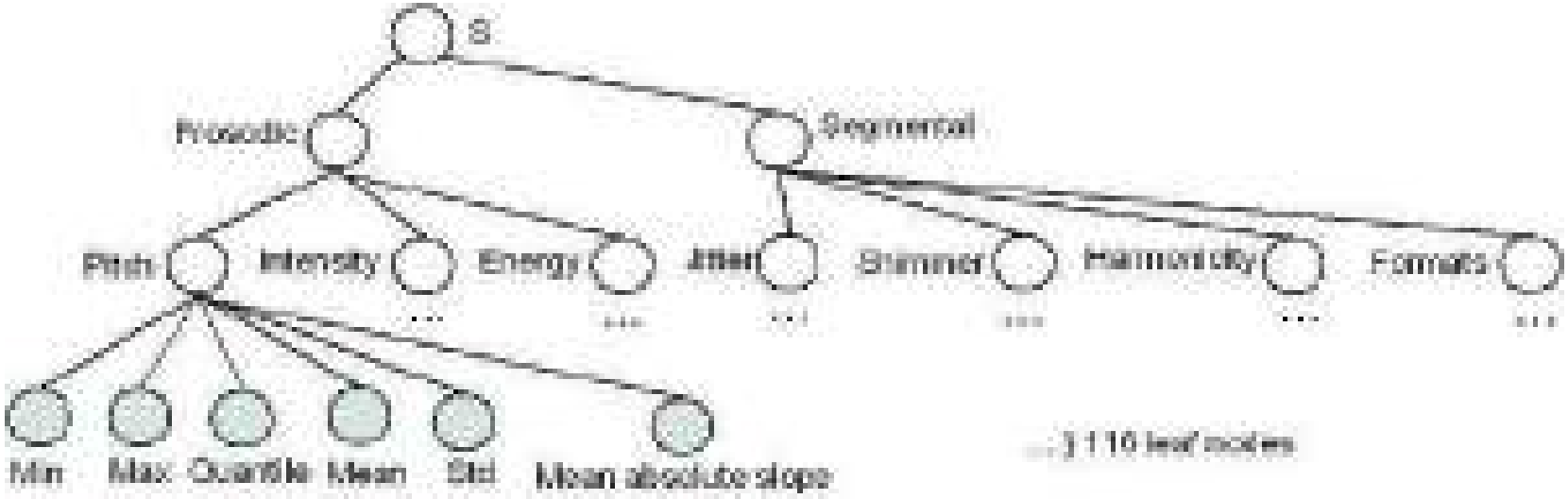


TGI+.1: How sample is classified?

A new input sample is fed to TGI+ in the form of a 116 dimensional feature vector.

1. Edit distances from the sample to seven tree automata are calculated.
2. The C4.5 decision tree on distance-to-automaton values is used to classify the sample. The tree was learnt at the training stage.

Model speech sample with tree



My Grammar Inference Algorithm

Automaton per class.
From examples learn grammar for each automaton:

1. learn the skeleton from the first sample;
2. for each leaf, learn the numeric intervals;

My Edit Distance Calculation Algorithm

$$D_V = 0, \tag{1}$$

in case $x \in [I_L, I_R]$.

$$D_V = \frac{I_L - x}{I_R - I_L}, \tag{2}$$

in case $x \leq I_L$.

$$D_V = \frac{x - I_R}{I_R - I_L}, \tag{3}$$

in case $x \geq I_R$.

The cost for every upper node $= \sum$ costs of its ancestors;

My extension: weights

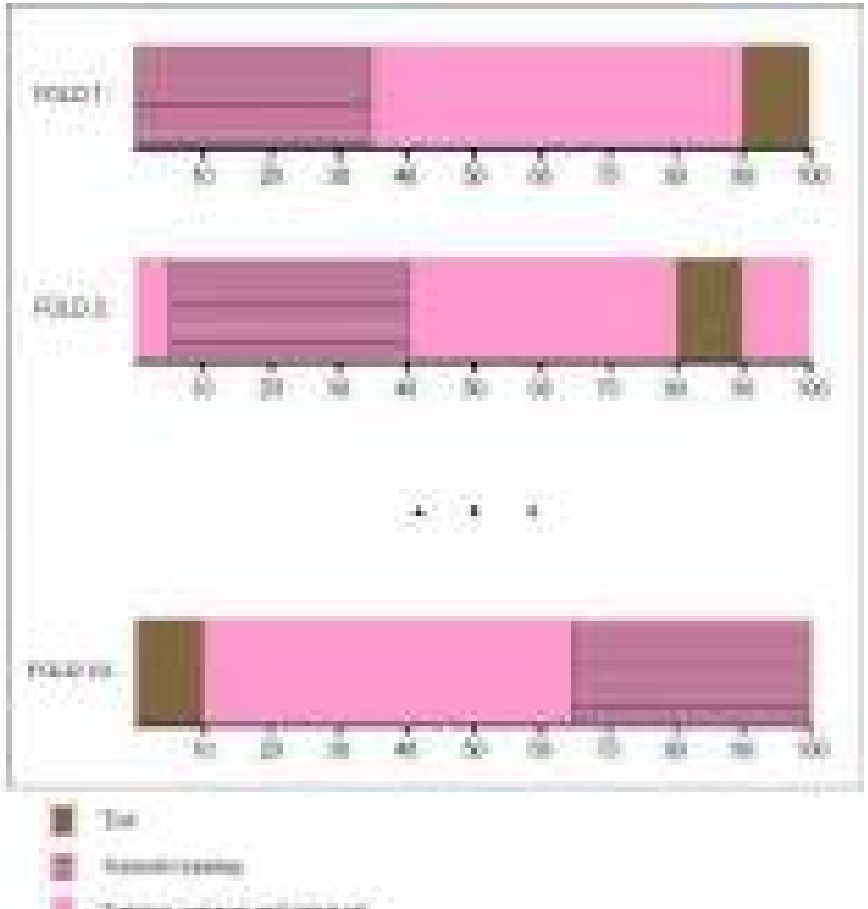
enhance the tree grammar inferece with a statistical wrapprer
feature selection:
e.g. the correlation based feature selection
class i vs rest

- either treat selected and non selected features equally
- or put a lot more of weight on selected features

$k = \frac{w_n}{w_s}$ on [0,1].
$f(k)$: f is accuracy on the validation set
the best $(w_n, w_s) = (1; 1.5)$.

TGI+: experiments

dataset: acted, German, benchmark EMO-DB
emotions: fear, disgust, happiness, boredom, neutral, sadness, anger
competitor: Multilayer Perceptron (the weka's top performer)
protocol: 10-F cross-validation



TGI+: results

1. baseline: TA = 43%
2. baseline: C4.5 = 52.9%
3. state of the art: MLP = 73.9%
4. TGI+.2: 78.6%
   - I have arrived to a meaningful combination of pattern recognition paradigms
   - outperforms the state of the art classifier by 4.6% $\pm$ 3.5%, i.e. a statistically significant improvement

TGI+: the main property

a human-readable classification method

1. simple modeling: whether the feature fits an interval;
2. a human-readable decision tree based on the closeness to prototypical expressions of other basic emotions.
   - a potential aplication as a training tool for the patients with an impaired capability to express speech emotions

THANK YOU!

Lexical and Morphological Statistics of an Arabic POS-Tagged Corpus

Hamdy S. Mubarak **Kareem A. Shaban** **Forat M. Adel**

Arabic NLP Researches, Sakhr Software
Sakhr Building, Free Zone, Nasr City 11711, Cairo
Egypt
{hamdys,kshaban,forat}@sakhr.com

Abstract

Part-Of-Speech (POS) tagging is a basic component necessary for many Natural Language Processing (NLP) applications. Building a manually tagged corpus helps in studying key statistics of a given language which form the basis for POS tagging systems. In this paper, we present both lexical and morphological statistics for Arabic that are derived from the Sakhr's POS manually tagged corpus. It covers text (7 M words) from a wide range of Arab countries in different domains over the years 2002-2004. The derived statistics are used as heuristics and preferential rules within a statistical Diacritizer which achieves a high accuracy in stem diacritization and POS disambiguation. Statistics includes information related to sentence and word lengths, punctuation marks, distribution of Arabic letters and diacritics, in addition to lexical and morphological information for POS distribution, stems, prefixes, suffixes, roots, morphological patterns, and morphosyntactic features like gender, number, person, and case ending. Modern Standard Arabic (MSA) is studied by analyzing the coverage of stems, roots, morphological patterns, prefixes, and suffixes. Comparisons with an arbitrary English corpus are shown in applicable cases.

Keywords: Corpus Statistics, Arabic NLP, POS Tagging, Diacritization, MSA

1. Introduction

Part-Of-Speech (POS) tagging is assigning a specific tag to each word of a sentence to indicate its function in the specific context [1]. POS tagging is considered as one of the basic components necessary for any robust Natural Language Processing (NLP) infrastructure [2], and it is needed in many tasks such as syntax and semantic analysis, text to speech (TTS), natural language parsing, information retrieval (IR), information extraction (IE), and machine translation (MT) [3].

A manually tagged corpus can be used for innumerable studies of word-frequency and POS. It also inspires the development of similar "tagged" corpora. Statistics derived by analyzing such corpus formed the basis of the latest POS tagging systems.

In this paper we will describe many lexical and morphological statistics that are derived from Sakhr's Arabic manually POS-Tagged corpus (POST) hand tagged by human annotators. These statistics include POS distribution, usage of stems, prefixes, suffixes, roots, morphological patterns, and also the usage of morphosyntactic features like gender, number, person, case ending, etc.

The benefits of these statistics were gained when they are considered as heuristics and preferential rules while building a Statistical Diacritizer which successfully disambiguates Arabic sentences by selecting the appropriate morphological analysis including POS, stem diacritics and morphosyntactic features. This Diacritizer also suggests the final case ending for each word which represents the syntactic function of words in context.

A comparison between Arabic and English corpora is conducted which considered some aspects like sentence length, word length, unique words, and punctuation marks. As a matter of fact, POST had a significant impact on training the statistical diacritizer's models whose stem diacritization and POS disambiguation accuracy reached 97%, and final case ending diacritization reached 92%.

This paper is organized as follows: Section 2 is a brief introduction to some aspects of Arabic language. Sections 3 and 4 describe Sakhr's morphological analyzer and POST. Section 5 through 17 present detailed language statistics. Finally, section 18 gives some concluding remarks.

2. Aspects of the Arabic Language

Arabic is one of the six official languages of the United Nations and the mother tongue of more than 300 million people. It is the official language in 25 countries (also widely studied and used throughout the Islamic world), and the third most after English and French. Arabic is the largest living Semitic language whose main characteristic feature is that most words are built up from roots by following certain fixed morphological patterns (which specify the vowels that can follow each consonant of root letters) and adding infixes, prefixes and suffixes. Arabic includes 28 letters and it is written cursively from right to left [4]. Arabic morphology is rather complex because of the morphological variation and the agglutination phenomenon. Letters change forms according to their position in the word (beginning, middle, end and separate) [5].

The modern form of Arabic is called Modern Standard Arabic (MSA), which is a simplified form of Classical Arabic, and it is the form used by all Arabic-speaking countries in publications, workplaces, government and media [6]. MSA is very often written without diacritics, which leads to a highly ambiguous text. Arabic readers could differentiate between words having the same writing form (homographs) by the context of the script [7].

3. Morphological Analysis

Sakhr's Morphological Analyzer is a morphological analyzer-synthesizer that provides basic analyses of a single Arabic word, covering the whole range of modern and classical Arabic. For each analysis, it provides its morphological data such as stem, root, morphological pattern, POS, prefixes, suffixes and also its morphosyntactic features like gender, number, person, case ending, etc. In addition to its high accuracy (99.8%), the Morphological Analyzer sorts the word analyses according to the usage frequency (using manual ordering of analyses for commonly-used words as appeared in an Arabic corpus of 4G words, or ordering according to stem frequency, otherwise). This morphological analyzer is integrated in most Sakhr products like TTS, MT, Search Engine and Text Mining.

4. Arabic POS-Tagged Corpus

POST includes texts (from newspapers, news services, and magazines) from different Arabic-speaking countries in different domains (Politics, Economy, Sport, Religion, Science, Medicine, etc) over the years 2002-2004. The corpus size is about 7M words (~330K sentences).

In our study of Arabic spelling mistakes in newspapers, we found out that Common Arabic Mistakes (CAM) occur in initial Hamza, final Taa Marbuta, and final dotted Yaa with a percentage varying from 1% to 12%, with an average of 5% of words. So, preprocessing of Arabic text is necessary, before tagging process takes place, in order to correct and normalize Arabic text by removing diacritics and irrelevant characters.

For each word in a sentence and based on its surrounding context, human annotators select the appropriate morphological analysis from all analyses generated by the Morphological Analyzer for this word, and also determine the final case ending based on this context. Out-Of-Vocabulary (OOV) words and wrong analyses are also flagged during the tagging process and this gave a great feedback to the lexicon, proper nouns, and corrector databases.

For a comparison with an English corpus, we selected texts with same size (7M words) from famous news agencies.

Figure 1 shows the sentence length distribution in both Arabic and English corpora. The average length of sentence is 21 words in Arabic and 19 in English. In 95% of the cases, sentence length is in the range 2-37 words in Arabic and 2-42 in English.

5. Sentence and Word Lengths

On the other hand, Figure 2 shows the word length distribution (in characters) in Arabic and English. The average length of word is 5 Characters in Arabic and 3 in English.

In 95% of the cases, word length is in the range 2-9 characters in Arabic and 2-11 in English.

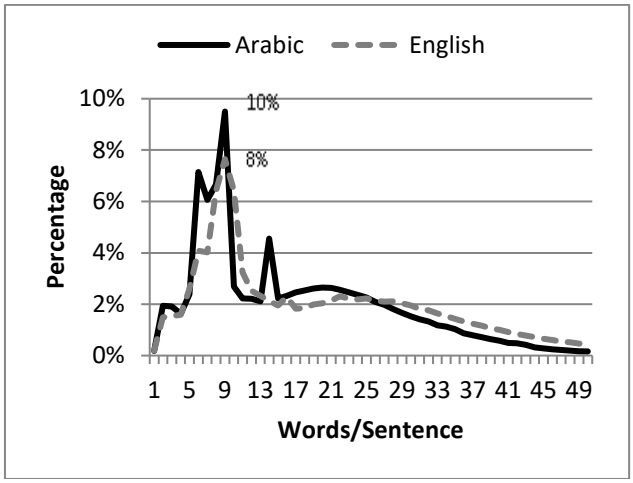


Figure 1: Sentence Length Distribution in Arabic and English

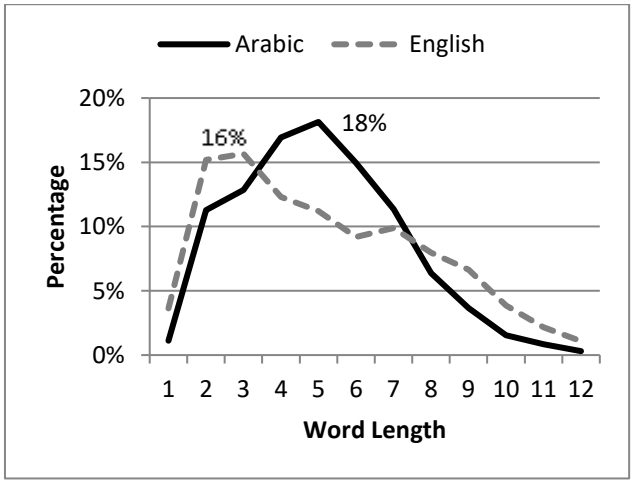


Figure 2: Word Length Distribution in Arabic and English

6. Arabic Letters

Figure 3 shows the distribution of Arabic letters. It is notable that, in any Arabic document, only 2 letters ("ا *A*" and "ل *l*") [1]represent 26% of the existing letters, and 6, represent 50%. These 6 letters are ("ا *A*", "ل *l*", "ي *y*", "م *m*", "ن *n*" and "و *w*") and they are used in the definite article ("ال *Al*"), long vowels ("ا *A*", "و *w*" and "ي *y*"), and the letters ("م *m*" and "ن *n*") that are frequently used in some function words and commonly in others.

[1] Buckwalter Arabic transliteration scheme (http://www.qamus.org/transliteration.htm) is used in all applicable cases.

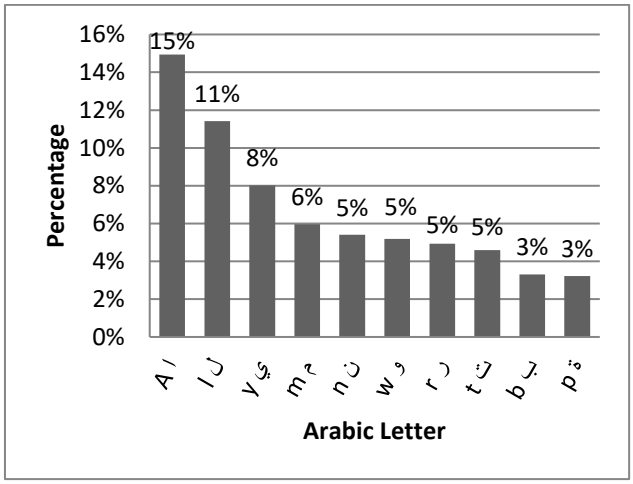


Figure 3: Most Frequent 10 Letters

7. Unigrams in Arabic and English

Unigrams represent how frequent a certain token has been written in a corpus. Arabic has a larger number of unigrams because Arabic has a very rich and complex morphology than English [7]. Moreover, the concatenation of affixes (prefixes and suffixes) with stems generates new unigrams. Figure 4 shows the distribution of unique words (unigrams).

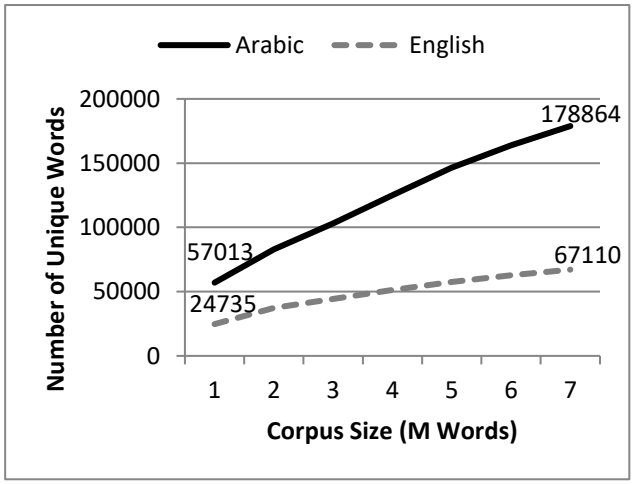


Figure 4: Number of Unique Words in Arabic and English

Table 1 shows the most frequent 20 words in Arabic and English corpora in addition to the percentage of appearance. It is observed that the majority of these words are function words (prepositions represent ~9%) and have no direct relation with the idea of the document. However, they play a significant role in binding words together.

Arabic		English	
word	**%**	**word**	**%**
fy في	3.55	the	5.1
mn من	2.09	of	2.59
>n أن	1.4	in	2.36
ElY على	1.4	to	2.18
<lY إلى	1.06	and	1.9
<n إن	0.61	a	1.38
En عن	0.58	that	1.25
Alty التي	0.54	for	0.73
wqAl وقال	0.41	on	0.73
mE مع	0.4	The	0.57
*Al*y* الذي	0.36	is	0.57

bEd بعد	0.29	with	0.51
*h*h* هذه	0.28	said	0.47
byn بين	0.26	by	0.42
qd قد	0.25	as	0.4
*h*A* هذا	0.24	was	0.38
lA لا	0.24	it	0.36
mA ما	0.23	from	0.35
lm لم	0.18	an	0.31
>nh أنه	0.18	not	0.31

Table 1: Most Frequent 20 Words in Arabic and English

8. Punctuation Marks

One of the most useful features in detecting sentences boundaries and tokens is punctuation marks. Unfortunately, writers do not pay attention to punctuation marks usage in Arabic, and they are considered by some as redundant cosmetic marks [7]. Figure 5 shows punctuation marks distribution in Arabic and English. It is remarkable that Arabic documents are full of inconsistent styles of punctuation marks like two consecutive commas, mixing of single and double quotations, two consecutive question marks, and incorrect representation of period as a zero digit.

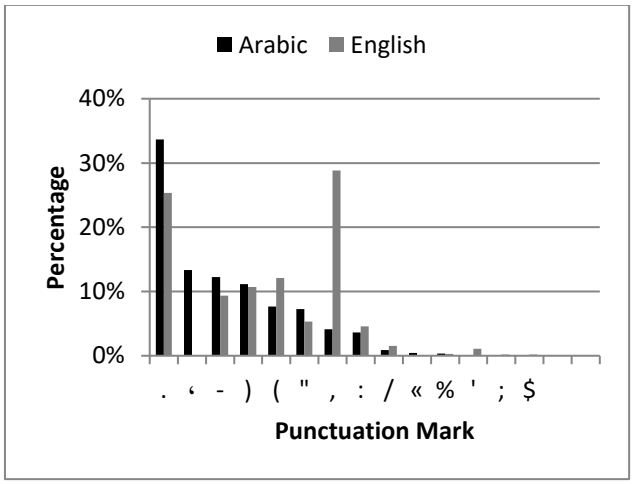


Figure 5: Punctuation Marks in Arabic and English

9. MSA Ambiguity

Short vowels are indicated by diacritics and are very often omitted from the modern writing style. It can be easily observed that MSA tends to be simpler than the Classical Arabic in grammar usage, syntax structure, morphological and semantic ambiguity. This will help normal Arabic readers to understand the written text. For example, 69% of words in the Arabic corpus have only 1 identified morphological analysis (one morphological interpretation), and 19% have 2 analyses, while high ambiguous words (3+ analyses) represent 12% only as shown in Figure 6.

Because Sakhr's Morphological Analyzer provides an ordered list of analyses according to usage frequency, it was discovered that 92% of words occupy the first position in analyses, and 5% occupy the second one as shown in Figure 7, which means that MSA in most cases is not so ambiguous, and words occupy the "trivial" analysis. For example, the word "للحاكم *llHAkm*" has more than one analysis (لِلْحَاكِمِ *liloHaAkimi*, to/of/for the ruler, لِلِحَاكُمْ *liliHaAkumo*, to/of/for your beards, etc), but the first one is usually recognized.

Figure 8 shows the relation between the word length and its morphological ambiguity (number of analyses). On the average, an Arabic word has 1.5 analyses, and in the extreme cases when length of word is too short (1 character) or too long (15+ characters), it tends to have only one analysis.

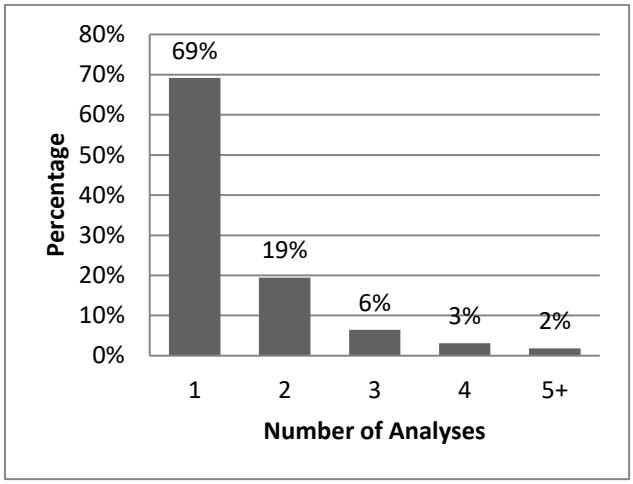


Figure 6: Distribution of Number of Analyses

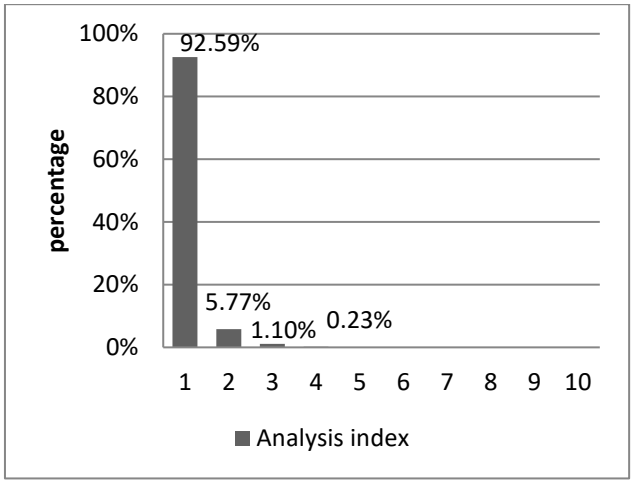


Figure 7: Distribution of the Selected Analysis Index

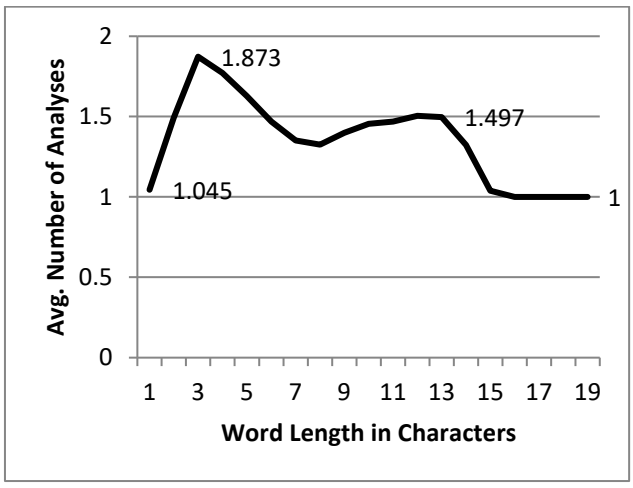


Figure 8: Morphological Ambiguity and Word Length

10. POS Distribution

Arabic grammarians traditionally analyze all Arabic words into three main parts of speech or categories, which are further sub-categorized and collectively cover the whole Arabic language [6]. These parts are: **Noun** (a name or a word that describes a person, thing, or idea), **Verb** (a word that denotes an action), and **Particle** (anything else, includes prepositions, adverbs, conjunctions, interrogative particles, exceptions, and interjections). Figure 9 shows the POS distribution after manual POS disambiguation of the Arabic Corpus.

It is notable that nouns represent 62% of POS, verbs represent 10%, while particles represent 28%. In addition, the usage of imperative verbs and passive voice of past and present verbs is rare in MSA (less than 1%), and they are usually replaced by less ambiguous words and structures. For example, instead of writing the ambiguous passive verb in the sentence "أُفتُتح المشروع *AfttH Alm$rwE*" (was-inaugurated the-project), another simple structure is used "تم افتتاح المشروع *tm AfttAH Alm$rwE*" (has-been inaugurating the-project).

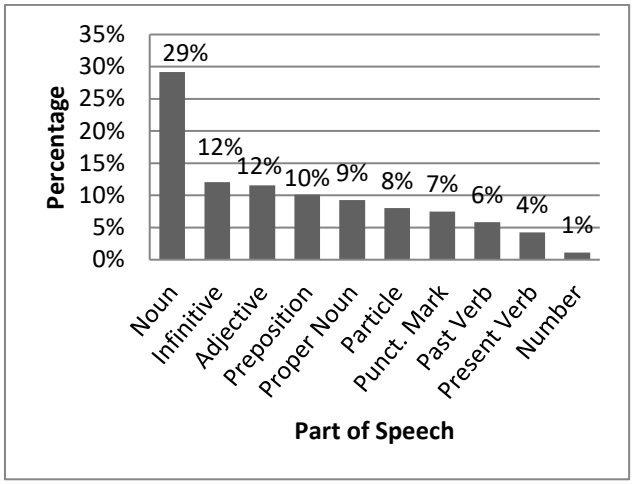


Figure 9: Most Frequent 10 POS's

In the following sections, we will describe some of the lexical and morphological statistics that are derived from POST after assigning each word in a sentence to its appropriate morphological analysis based on its context. The morphological analysis includes information about stem (which is divided more into root and morphological pattern), affixes (prefixes and suffixes), and morphosyntactic features (like the gender, number, person, case ending, etc.)

11. Stem Distribution

Most Arabic words are morphologically derived from a list of roots; it can be tri-, quad-, or pent-literal. Most of these roots are tri-literal. Arabic words may have no root (for the majority of function words, some of proper nouns and borrowed words). Figure 10 shows the distribution of root types. This figure shows that quad-literal roots are rarely used in MSA.

Figures 11 and 12 show the most frequent roots, and morphological patterns, respectively. The most frequent roots used during this period of time were "رءس *r's*" and "عرق *Erq*" because of the events that were happening in " العراق *AlErAq*, Iraq" and their effect on most of the publications and media. The most frequent morphological patterns are both "فَعْل *faEol*" which represents the noun, and infinitive, and "فَاعِل *faAEil*" which represents the adjective in most cases.

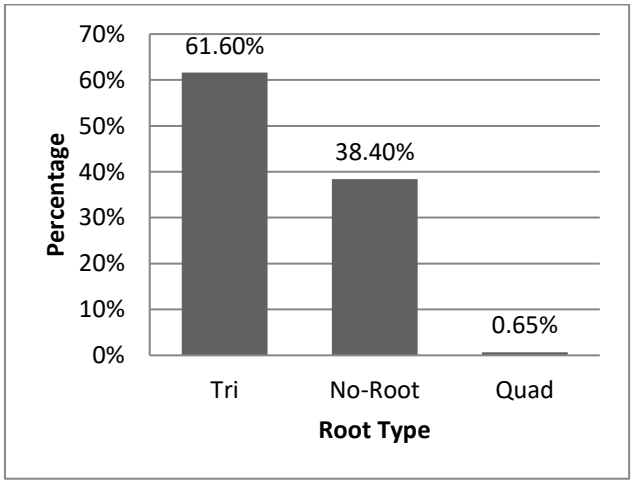


Figure 10: Root Type Distribution

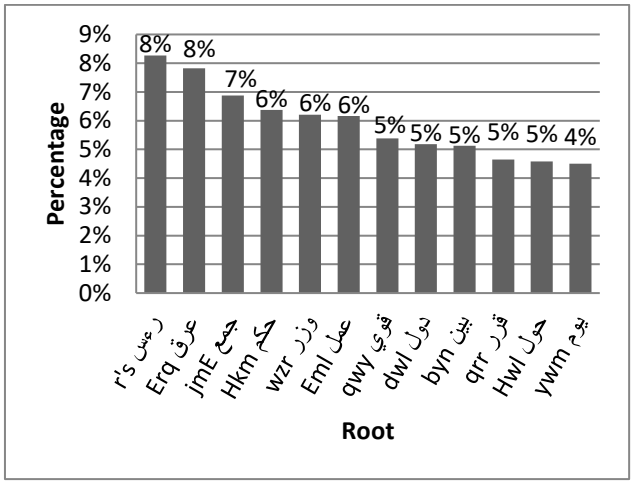


Figure 11: Most Frequent Roots

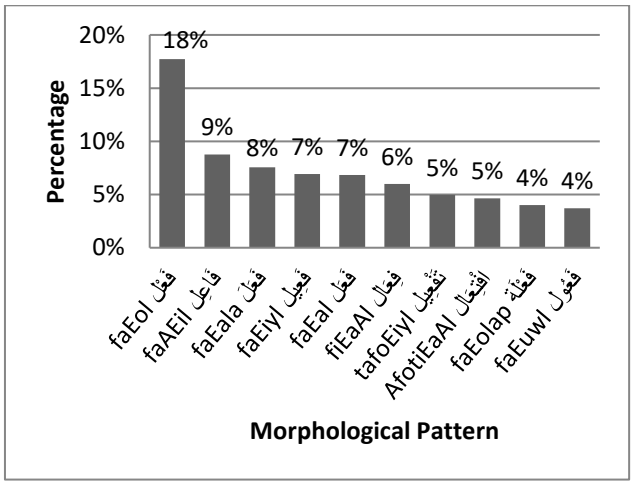


Figure 12: Most Frequent 10 Morphological Patterns

12. Affixes Distribution

Affixes (prefixes and suffixes) are agglutinated to the beginning and the end of Arabic words. Prefixes are generally conjunctions, prepositions, and determiners (and include also the person conjugation of verbs in the present tense "أنيت" حروف المضارعة). Suffixes are the conjugation terminations of verbs and they are the dual/plural/feminine marks for nouns, and pronouns attached at the end of words [5].

Figures 13 and 14 show the distribution of prefixes and the conjugation person of present verbs.

We can observe that most words have no prefixes (87%), and 12% have only 1 prefix ("و *w*", "ب *b*", or "ل *l*"), while other prefixes are rarely used.

On the other hand, Figure 15 shows the suffixes distribution, and it is notable that 76% of words have no suffixes, and 17% have simple ones, while other suffixes are rarely used.

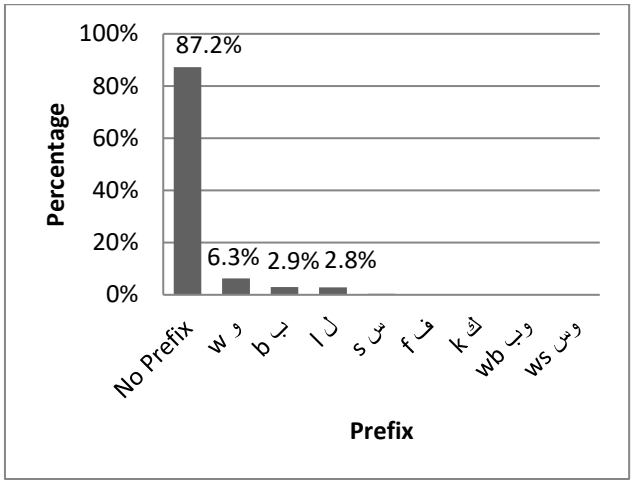


Figure 13: Most Frequent Prefixes

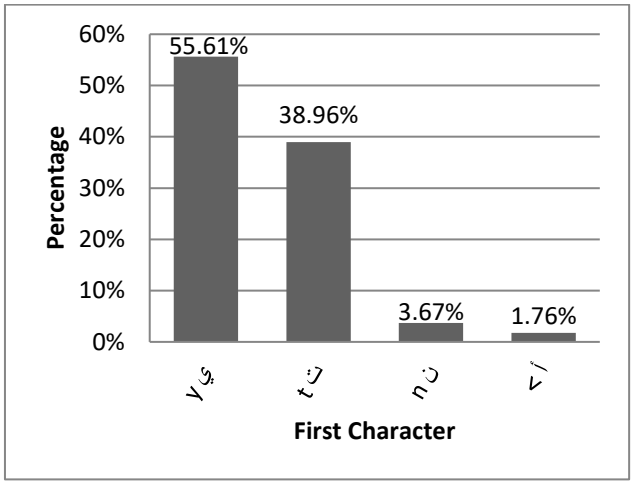


Figure 14: Person Conjugation of Present Tense

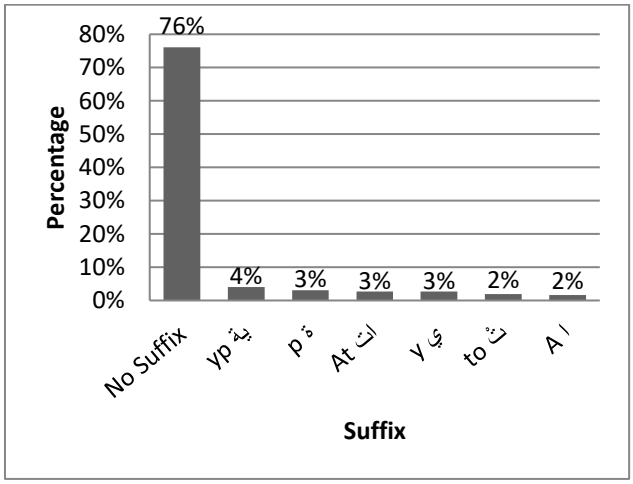


Figure 15: Most Frequent Suffixes

13. Morphosyntactic Features

In this section we show the distribution of *gender*, *number*, *person*, *case ending*, and *definiteness*.

Gender النوع in Arabic can be masculine, feminine, or neuter (like function words). Figure 16 shows the distribution of gender. It is notable that masculine words are more frequent than feminine words (1.5 times).

Number العدد in Arabic can be singular, dual, or plural (plural is divided more into regular plural and broken plural). Figure 17 shows the distribution of number. It is notable that singular words are more frequent than plural words, while using dual number is very limited (~5%).

Person الشخص in Arabic can be first person (narrator متكلم), second person (interlocutor مخاطب), or third person (absent غائب). Because of the narrative nature of most of Arabic publications (especially newswire and media), the third person is dominant (~97%) while second and first persons are almost equal as shown in Figure 18.

Case Ending الحالة الإعرابية for nouns can be nominative مرفوع, accusative منصوب, genitive مجرور, or given مبني (fully diacritized without considering the case ending mark), while the case ending for verbs can be indicative مرفوع, subjunctive منصوب, jussiveمجزوم , or given مبني . Examples for given nouns are particles, and pronouns, and for given verbs are past verbs, imperative verbs, and present verbs with some suffixes.

Figure 19 shows the distribution of case ending for nouns and verbs. We can observe that the case ending for verbs (if not given) tends to be indicative (~81% of the cases), and for nouns (if not given) it tends to be genitive (~56% of the cases).

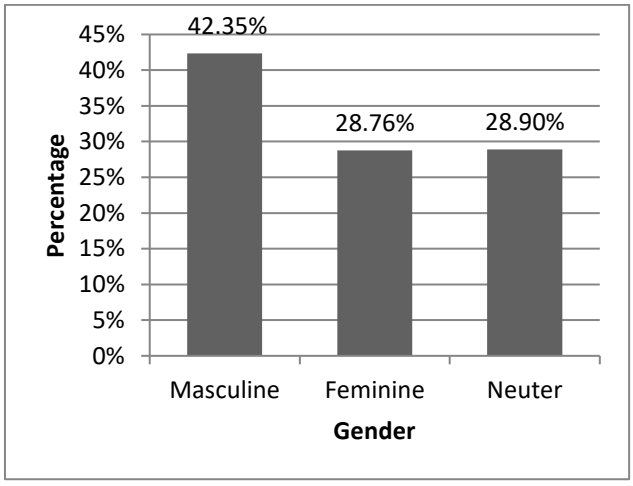


Figure 16: Gender Distribution

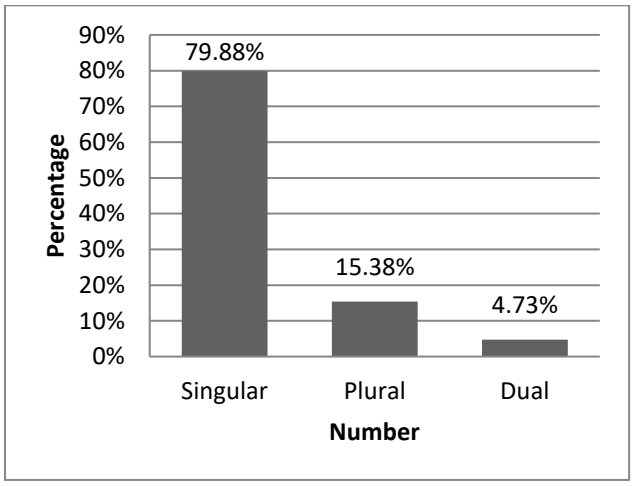


Figure 17: Number Distribution

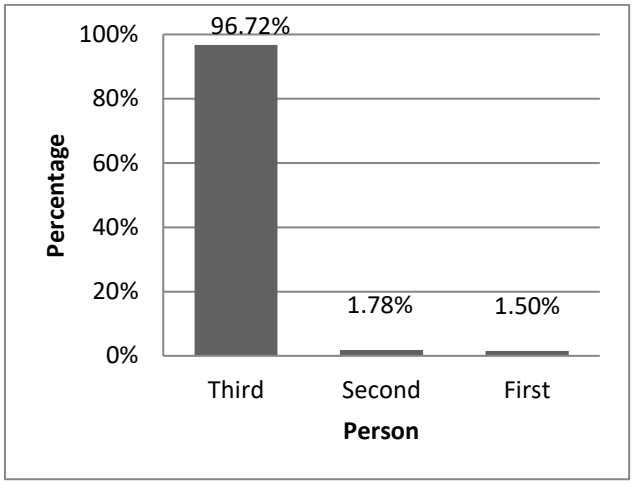


Figure 18: Person Distribution

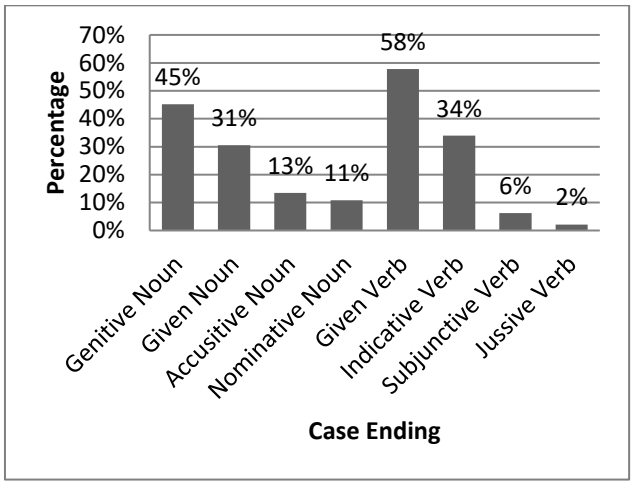


Figure 19: Case Ending Distribution

Figure 20 shows the distribution of diacritics extracted from the fully diacritized corpus. It is notable that "Fatha" is the most frequent diacritic and forms with "Kasra", "Sukun" and "Damma" ~97% of the whole diacritics.

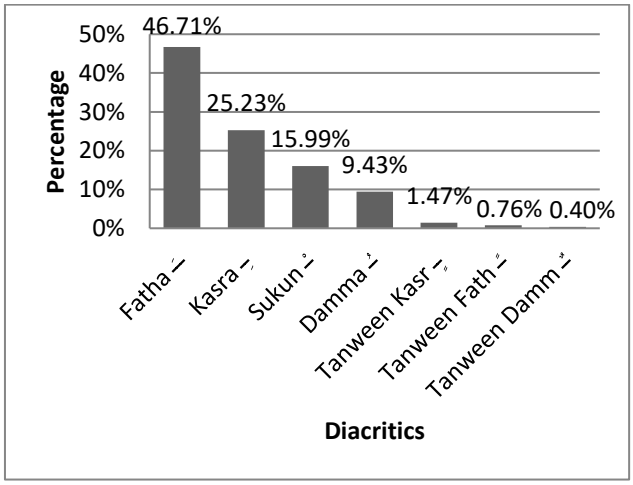


Figure 20: Diacritics Distribution

Definiteness التعريف in Arabic can be definite with the definite article AL معرف بال, definite without AL معرف بغير ال (like proper nouns, pronouns, and in possessive pronouns suffixes cases), or indefinite نكرة as in Figure 21.

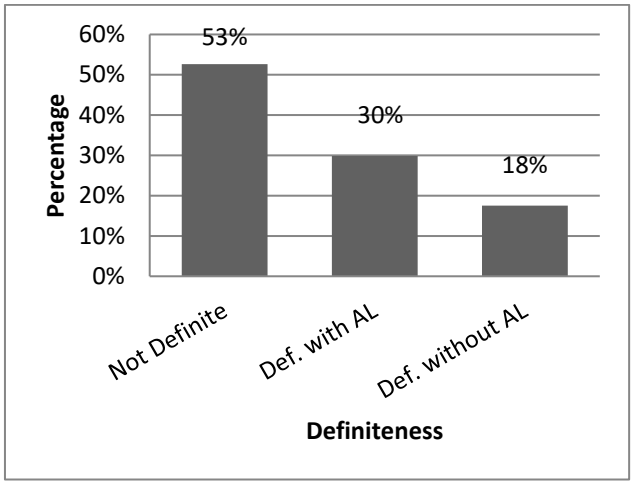


Figure 21: Definiteness Distribution

14. Corpus Coverage

In this section we discuss the coverage of existing unique words in POST and compare it with an arbitrary recent corpus that is crawled by Sakhr's news gathering service (Johaina http://johaina.sakhr.com) which gathers Arabic text from more than 400 Arabic sources. The objective of this comparison is to answer the following question: If we have an arbitrary recent corpus, what are the differences between our "old" tagged corpus and this new one in terms of new unique words, new stems, and new proper nouns?

To study the unique words coverage, we gathered a recent corpus from Johaina with a size of 14M words (double POST size), and normalized tokens in both corpora (to exclude mismatches due to spelling mistakes in the crawled corpus and POST corrected corpus). Out of 172K normalized unique words in POST and 298K normalized unique words in Johaina, there was an intersection of 124K words which represents 73% of POST and 42% of Johaina as shown in Figure 22.

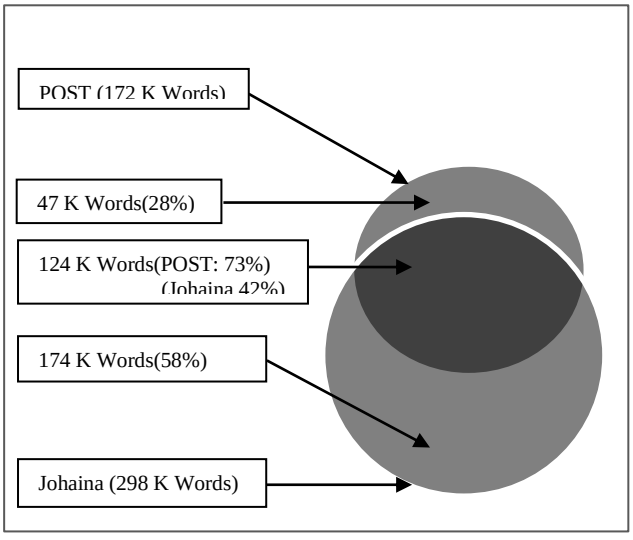


Figure 22: Unique Words Coverage

When we analyzed the words that are found in POST but not found in Johaina and vice versa, we observed the following:
- Missing stems in POST (with affix expansion) represent 11% of these words which indicate new stems in MSA or uncovered ones in POST like: "حلحلة *HlHlp*, حوكمة *Hwkmp*, and تمدرس *tmdrs*", while missing stems in Johaina represent 2% of these words like: "فدائي *fdA}y*, مستنسخ *mstnsx*, and تسلحية *tslHyp*" that are no longer mentioned extensively in modern writings as obtained from Johaina corpus.
- Stems with different affixes and obsolete/new proper nouns represent 98% and 87% of POST and Johaina stems in order, as shown in Figure 23.

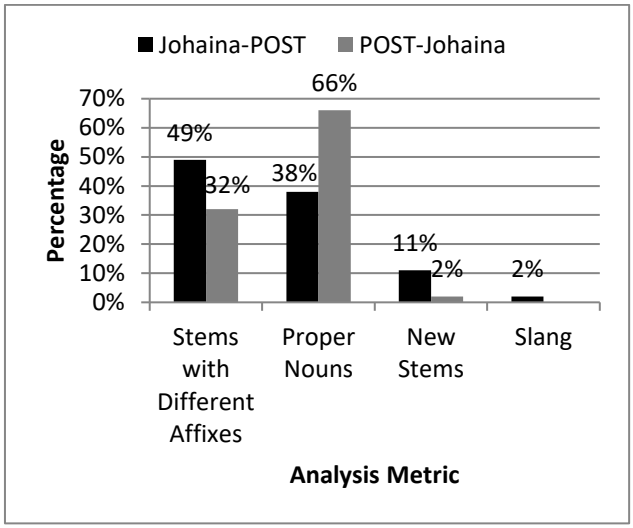


Figure 23: Analysis of Uncovered Stems

15. MSA Morphological Coverage

The morphological analyzer uses the lexical database (LDB) to analyze and synthesize Arabic words. LDB contains lists of stems, roots, morphological patterns, prefixes, and suffixes, etc., as mentioned in common Arabic lexicons and resources (like المعجم الوسيط and المعجم العربي الأساسي).

In this section we study the coverage of these morphological data that appeared in our tagged corpus with respect to the corpus size. For any of the next information, we consider a single existence of any morphological data value as covered, otherwise, we consider this value uncovered (unused).

For **stem coverage**: Figure 24 shows the relation between the corpus size and existing stems. LDB contains 38,500 tri-literal stems, 1,200 quad-literal stems and 6,500 stems with no-root. For the whole corpus size (7M words), the coverage percentages of stems reached 52%, 39% and 57%, respectively. Examples of uncovered stems are: ميعاس myEAs, قاووق qAwwq، and تيهور tyhwr.

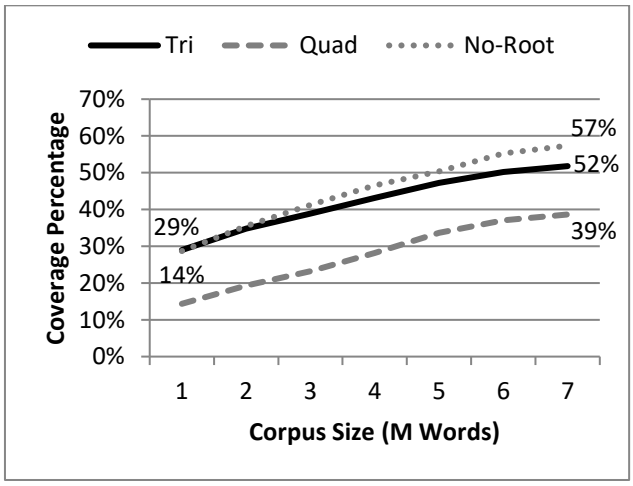


Figure 24: Stems Coverage Distribution

For **root coverage**: Figure 25 shows the relation between the corpus size and existing roots. LDB contains 5,000 tri-literal roots, and 800 quad-literal roots. For the whole corpus size (7M words), the coverage percentages reached 86% and 34%, respectively. Examples of unused roots are: kdn كدن, جدح jdH, and يفخ yfx.

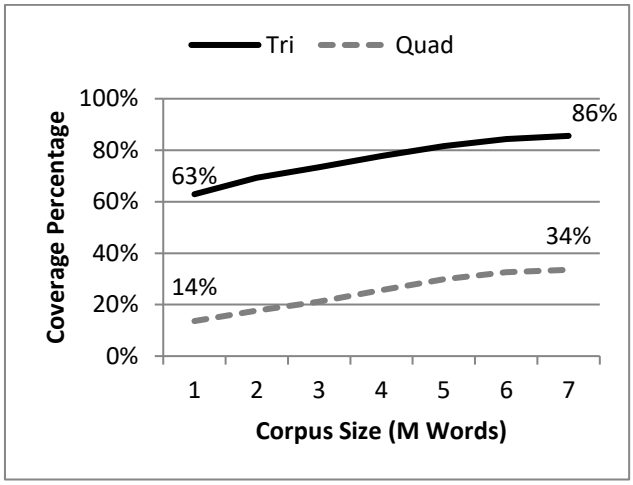


Figure 25: Roots Coverage Distribution

For **morphological patterns coverage**: Figure 26 shows the relation between the corpus size and the existing morphological patterns. LDB contains 540 tri-literal morphological patterns, and 110 quad-literal morphological patterns. For the whole corpus size (7M words), the coverage percentages were 55% and 46%, respectively. Examples of unused morphological patterns are: تَفَيْعَلَ tafayoEala, فِعْوَال fiEowaAl, and يُنْفَعَل yunofaEal.

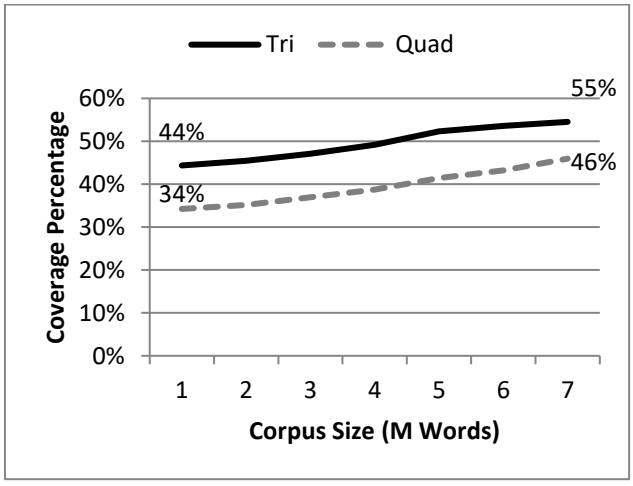


Figure 26: Morphological Patterns Coverage Distribution

For **prefixes coverage**: Figure 27 shows the relation between the corpus size and existing prefixes. LDB contains 140 Prefixes. For the whole corpus size (7M words), the coverage percentage was only 15%. Examples of unused prefixes are: أو >w, أس >s, and أوب >wb.

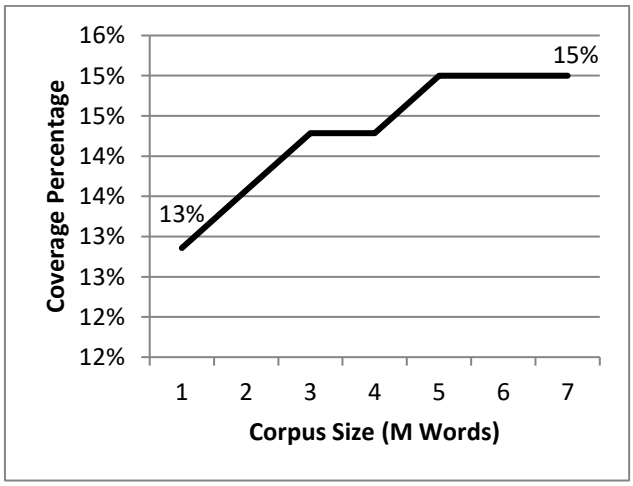


Figure 27: Prefixes Coverage Distribution

For **suffixes coverage**: Figure 28 shows the relation between the corpus size and existing suffixes. LDB contains 700 suffixes. The coverage percentage was 32%. Examples of unused suffixes are: كهن khn ،كهاء khA, and اكما AkmA.

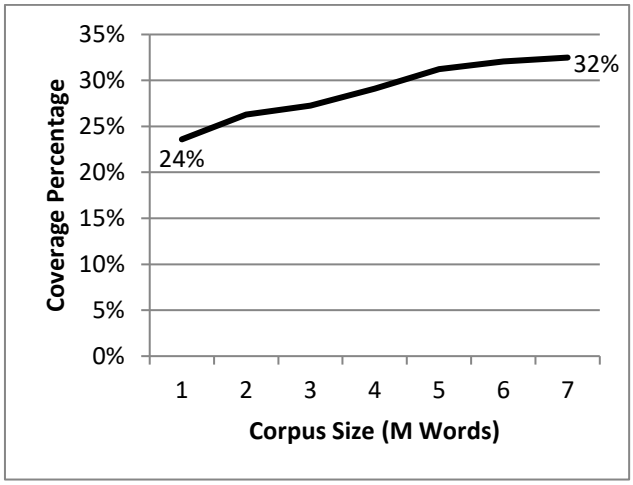


Figure 28: Suffixes Coverage Distribution

16. Other Annotated Corpora

Some previous attempts of Arabic corpora analysis are discussed in this section.

The Penn Arabic Treebank (PATB): Treebank is designed to support the development of data-driven approaches to NLP, human language technologies, automatic content extraction (topic extraction and/or grammar extraction), cross-lingual information retrieval, information detection, and other forms of linguistic research on MSA in general [8]. (http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004T11)

NEMLAR Arabic Written Corpus: aims to achieve a well-balanced corpus that offers a representation of the variety in syntactic, semantic and pragmatic features of modern Arabic language. The time span of the data included goes from late 1990's to 2005. The corpus is provided in 4 different versions: a) raw text, b) fully vowelized text, c) text with Arabic lexical analysis, and d) Arabic POS-tagged. (http://catalog.elra.info/product_info.php?products_id=873)

Prague Arabic Dependency Treebank (PADT): is a project of analyzing large amounts of linguistic data in Modern Written Arabic in terms of the formal representation of language that originates in the Functional Generative Description [9]. PADT does not only consist of multi-level linguistic annotations of the MSA, but it even has a variety of unique software implementations, designed for general use in NLP.
(http://ufal.mff.cuni.cz/padt/PADT_1.0/docs/index.html)

CLARA (Corpus Linguae Arabicae): The ultimate goal of this project is building a balanced and annotated corpus. The annotation is done for morphological boundaries and Part Of Speech (POS) [10].
(http://enlil.ff.cuni.cz/veda/projekty/clara.htm)

Table 2 shows some information about these corpora.

Corpus	Size (Words)	Years	Sources	Annotation
Sakhr	7 M	2002-2004	Different sources	POS+Morph
PATB	340 K	2000-2002	AFP, Al-Hayat, An Nahar	POS+Morph+Syntax
NEMLAR	500 K	1990-2005	Islamonline, RDI, An Nahar	POS+Morph
PADT	113 K	2000-2003	AFP,Ummah, An Nahar, Al-Hayat, Xinhua	POS+Morph+Syntax
CLARA	100 K	1997-1999	Different sources	POS+Morph

Table 2: Annotated Corpora Information

These annotated corpora use different morphological analyzers. At many levels, there are no standards. There are none for basic Arabic linguistic terms and their definitions, none for terms and their translation into English, and none for test collections and performance evaluations [11]. (Sakhr uses Sakhr's morphological analyzer, PATB and PADT use Buckwalter Arabic morphological analyzer (BAMA), while NEMLAR uses ArabMorpho© morphological analyzer).

17. Conclusions

In this paper, we presented lexical and morphological statistics of an Arabic POS-Tagged corpus and basic statistical differences between Arabic and English languages. Some useful statistics about the general characteristics (ambiguity, usage and coverage) of MSA were also obtained. In NLP applications, there is a new tendency to make use of statistical methods. The idea underlying this approach is observing how the language is actually used and drawing conclusions, instead of trying to formalize the language. The results given in this paper can be extended on this line. They are useful for statistical NLP approaches and different applications like Optical Character Recognition (OCR), spelling correction, POS disambiguation and diacritization, MT, IR, and IE.

18. References

[1] Jurafsky, D. and Martin, J.H. (2008). Speech and Language Processing: An Introduction to Speech Recognition, Computational Linguistics and Natural Language Processing. 2nd Ed., Prentice Hall, ISBN: 10: 0131873210.

[2] Atwell, E., Al-Sulaiti, L., Al-Osaimi, S. & AbuShwar, B. (2004). A review of Arabic corpus analysis tools. Proc. JEP-TALN'04 Arabic Language Processing.

[3] Elhadj, Y. (2009). Statistical Part-of-Speech Tagger for Traditional Arabic Texts. Journal of Computer Science 5 (11). Imam Muhammad Bin Saud University, KSA

[4] Seikaly, Z.A. (2007). The Arabic Language: The Glue that Binds the Arab World. AMIDEAST, America-Mideast Educational and Training Services, Inc. http://www.amideast.org/publications/arabic- language.pdf

[5] Kadri, Y., & Nie, J. Y. (2006). Effective Stemming for Arabic Information Retrieval. In Proceedings of the challenge of Arabic for NLP/MT Conference. The British Computer Society. London, UK.

[6] Khoja, S. (2001). APT: Arabic part-of-speech tagger. Proceeding of the Student Workshop at the 2nd Meeting of the NAACL, (NAACL'01), Carnegie Mellon University, Pennsylvania, pp: 1-6.

[7] Alotaiby, F., Alkharashi, I., & Foda, S. (2008). Processing Large Arabic Text Corpora: Preliminary Analysis and Results. King Saud University.

[8] Maamouri, M. & Bies, A. (2004). Developing an Arabic Treebank: Methods, Guidelines, Procedures, and Tools. In Proceedings of COLING 2004. Geneva, Switzerland.

[9] Hajič O. & et al (2006), The Challenge of Arabic For NLP/MT, Tips and Tricks of the Prague Arabic Dependency Treebank, International Conference at The British Computer Society (BCS), 23 October, London.

[10] Zemanek, P. (2001), CLARA (Corpus Linguae Arabicae): An Overview . In ELSNET (Ed.), Proceedings of ACL/EACL 2001 Workshop on Arabic Language Processing. Toulouse, France.

[11] Al-Sughaiyer, I., & Al-Kharashi, I. (2004), Arabic morphological analysis techniques: A comprehensive survey. Journal of the American Society for Information Science and Technology, 55(3):189–213.

HTK Tool Kit

HTK Tool Kit

What is HTK tool kit

The HTK language modeling tools are a group of programs designed for constructing and testing statistical *n-gram* language models

HTK Tool Kit

What to prepare

Training & Test Text

Dictionary

HTK Tool Kit

Training & Test Text

- Plain text sentences
- One sentence per line
- Sentence starts with <s>
- Sentence ends with </s>

HTK Tool Kit

Training Text Sample

<s> IT WAS ON A BITTERLY COLD NIGHT AND FROSTY MORNING TOWARDS THE END OF THE WINTER OF NINETY SEVEN THAT I WAS AWAKENED BY A TUGGING AT MY SHOULDER </s>

<s> IT WAS HOLMES </s>

HTK Tool Kit

Dictionary

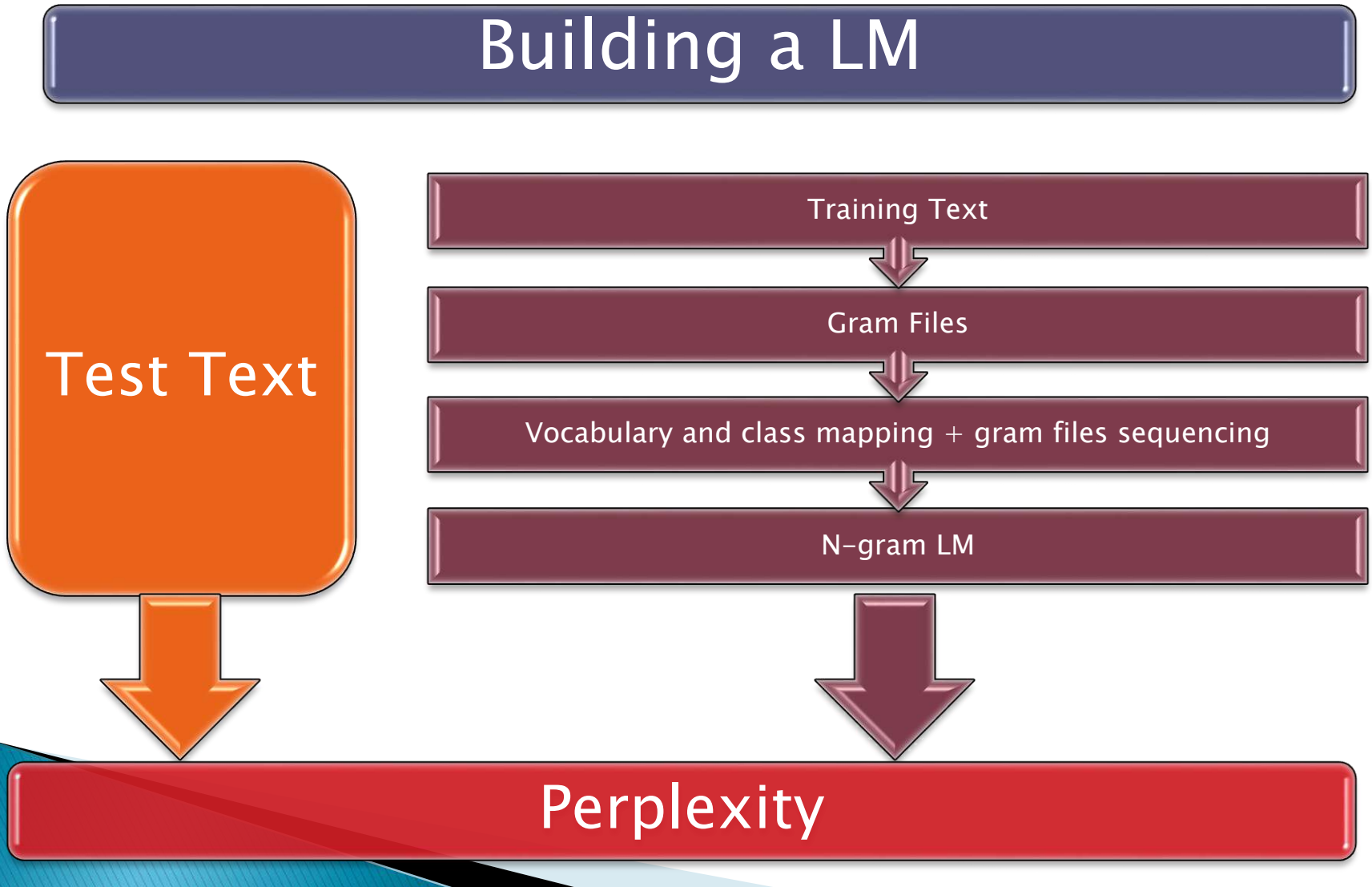
- Plain text wordlist
- One word per line
- Alphabetically ordered

HTK Tool Kit

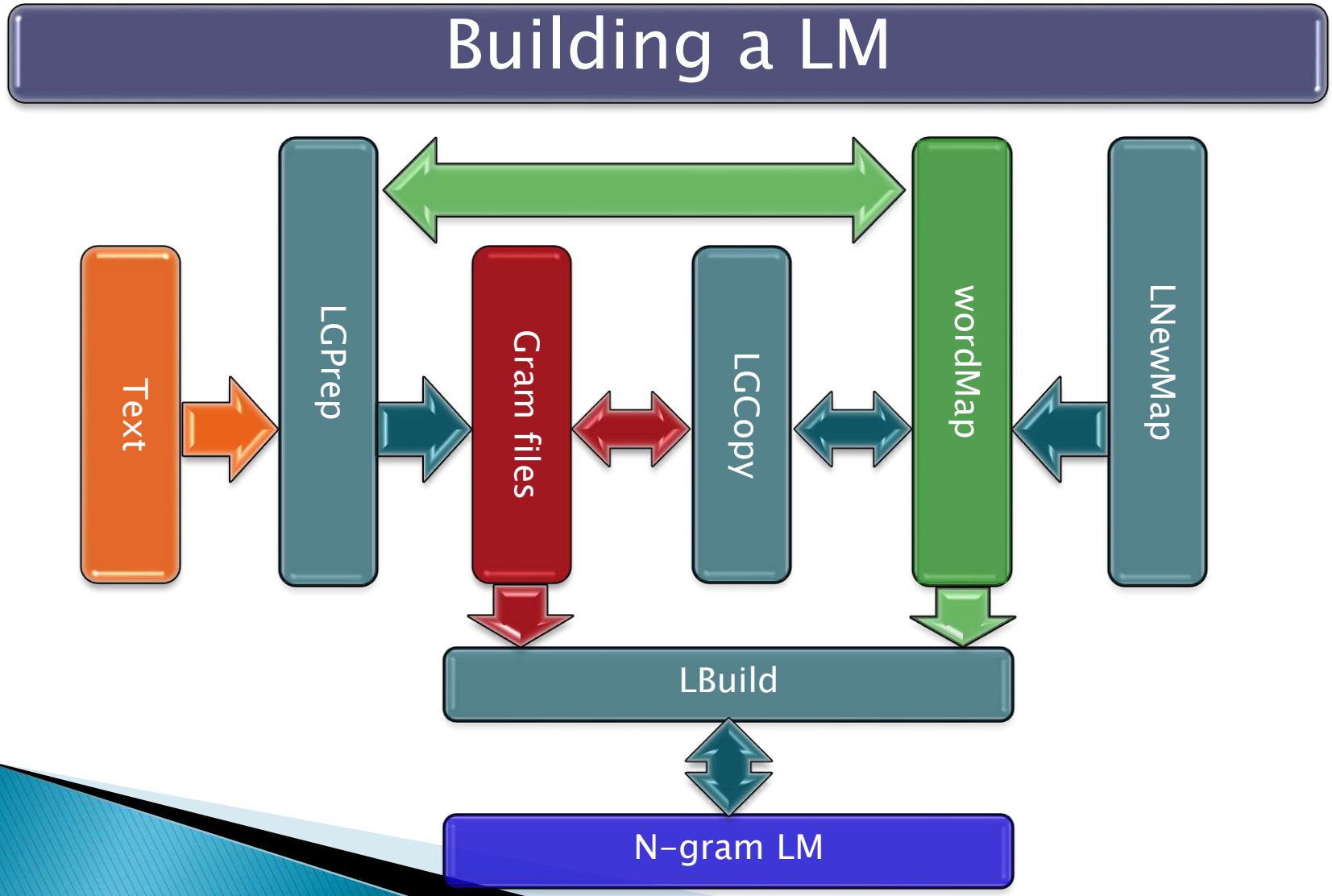
Dictionary Sample

</s>
<s>
A
A.
ABANDON
ABANDONED
ABBEY
ABDULLAH
ABE

HTK Tool Kit


Building a LM
Test Text
Training Text
Gram Files
Vocabulary and class mapping + gram files sequencing
N-gram LM
Perplexity

HTK Tool Kit


Building a LM
Text
LGPrep
Gram files
LGCopy
wordMap
LNewMap
LBuild
N-gram LM

HTK Tool Kit

LNewMap

LNewMap [options] name mapfn

-e esc Change the contents of the EscMode header to esc.
Default is RAW.

-f fld Add the field fld to the Fields header.

HTK Tool Kit

LNewMap

Example:

LNewMap -f WFC Holmes empty.wmap

```
Name = Holmes
SeqNo = 0
Entries = 0
EscMode = RAW
Fields = ID,WFC
\Words\
```

HTK Tool Kit

LGPrep

LGPrep [options] wordmap [textfile ...]

-a n Allow upto n new words in input texts (default 100000).

-b n Set the internal gram buffer size to n (default 2000000). LGPrep stores incoming n-grams in this buffer. When the buffer is full, the contents are sorted and written to an output gram file. Thus, the buffer size determines the amount of process memory that LGPrep will use and the size of the individual output gram files.

HTK Tool Kit

LGPrep cont'd

LGPrep [options] wordmap [textfile ...]

-d Directory in which to store the output gram files (default current directory).

-i n Set the index of the first gram file output to be n (default 0).

-n n Set the output n-gram size to n (default 3).

-r s Set the root name of the output gram files to s (default "gram").

HTK Tool Kit

LGPrep cont'd

LGPrep [options] wordmap [textfile ...]

-s s Write the string s into the source field of the output gram files. This string should be a comment describing the text source.

-z Suppress gram file output. This option allows LGPrep to be used just to compute a word frequency map. It is also normally applied when applying edit rules to the input.

HTK Tool Kit

LGPrep cont'd

Example:

LGPrep -T 1 -a 100000 -b 2000000 -d holmes.0 -n 4 -s "Sherlock Holmes" empty.wmap D:\train\abbey_grange.txt, D:\train\beryl_coronet.txt,...

HTK Tool Kit

LGPrep cont'd

WMAP file

```
Name = Holmes
SeqNo = 1
Entries = 18080
EscMode = RAW
Fields = ID,WFC
\Words\
<s>     65536  33669
IT      65537  8106
WAS     65538  7595
...
```

HTK Tool Kit

LGCopy

LGCopy [options] wordmap [mult] gramfiles

-b n Set the internal gram buffer size to n (default 2000000). LGPrep stores incoming n-grams in this buffer. When the buffer is full, the contents are sorted and written to an output gram file. Thus, the buffer size determines the amount of process memory that LGPrep will use and the size of the individual output gram files.

-d Directory in which to store the output gram files (default current directory).

HTK Tool Kit

LGCopy cont'd

LGCopy [options] wordmap [mult] gramfiles

-o n Output class mappings only. Normally all input *n*-grams are copied to the output, however, if a class map is specified, this options forces the tool to output only *n*-grams containing at least one class symbol.

HTK Tool Kit

LGCopy cont'd

Example:

LGCopy -T 1 -b 2000000 -d D:\holmes.1
D:\ holmes.0\wmap D:\ holmes.0\gram.1 D:\
holmes.0\gram.2.....

HTK Tool Kit

LBuild

LBuild [options] wordmap outfile [mult] gramfile ..

-c n c Set cutoff for n-gram to c.

-n n Set final model order to n.

HTK Tool Kit

LBuild cont'd

Example:

LBuild -T 1 -c 2 1 -c 3 1 -n 3 D:\lm_5k\5k.wmap
D:\lm_5k\tg2-1_1 D:\holmes.1\data.1
D:\holmes.1\data.2... D:\lm_5k\data.1 D:\lm_5k\data.12

HTK Tool Kit

LPlex

LPlex [options] langmodel labelFiles

-n n Perform a perplexity test using the n-gram component of the model. Multiple tests can be specified. By default the tool will use the maximum value of n available.

-t Text stream mode. If this option is set, the specified test files will be assumed to contain plain text.

HTK Tool Kit

LPlex cont'd

Example:

Lplex -n 3 -t D:\lm_5k\tg1_1 D:\test\red-headed_league.txt

ESEDA: tool for enhanced speech emotion detection and analysis

J. Sidorova
Group of Voice and Language,
Universitat Pompeu Fabra, Spain.
julia.sidorova@upf.edu

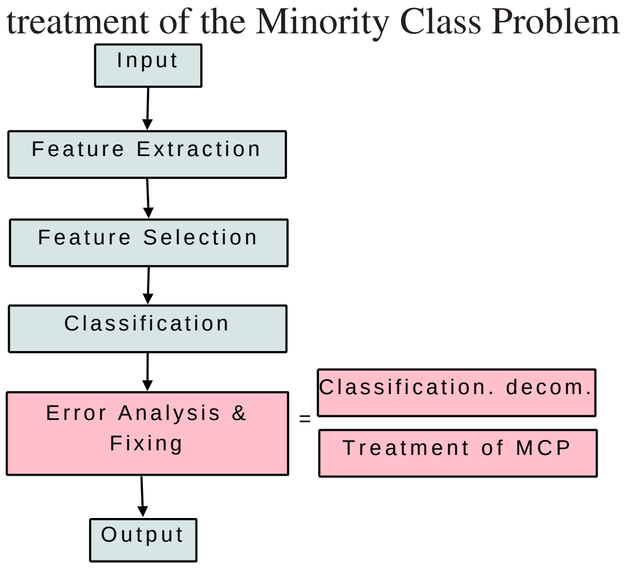
T. Badia
Group of Voice and Language,
Universitat Pompeu Fabra, Spain.
toni.badia@upf.edu

1 Introduction

An aim of a speech emotion recognition (SER) engine is to produce an estimate of the emotional state of the speaker given a speech fragment as input.

Figure 1: Motivation: HCI, etc [image taken from www.inf.ed.ac.uk/postgraduate/msc.html]

Figure 2: The standard way to do SER is through a supervised learning procedure (blue), we follow this trend and our contribution is an additional block based on error-analysis and fixing (pink). The block incorporates classification decomposition and treatment of the Minority Class Problem (MCP).



We do our experiments on the *Interface* database of acted emotional speech: 4 speakers, French.

2 Classification Decomposition

2.1 Preliminaries

Classification decomposition is splitting the complete multiclass problem into a set of smaller classification problems.
Advantages:

- Feature Selection is done for individual classification steps;
- Classification borders of smaller problems are usually simpler;

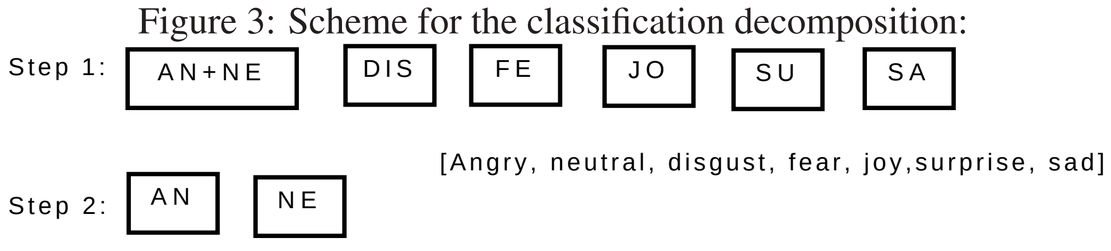
Therefore:

- better accuracy
- usually lower computational complexity

2.2 How Classification Path is Calculated?

1. class I: class of special interest [or the worst recognized class].
2. From the confusion matrix deduce:
   class J: the class with which class I is most frequently confused;
3. A new label K for Class I + Class J.
4. Classification step 1: recognize among all classes, where class K stands for I and J.
5. Classification step 2: class K $\longrightarrow$ class I or class J.

Figure 3: Scheme for the classification decomposition:



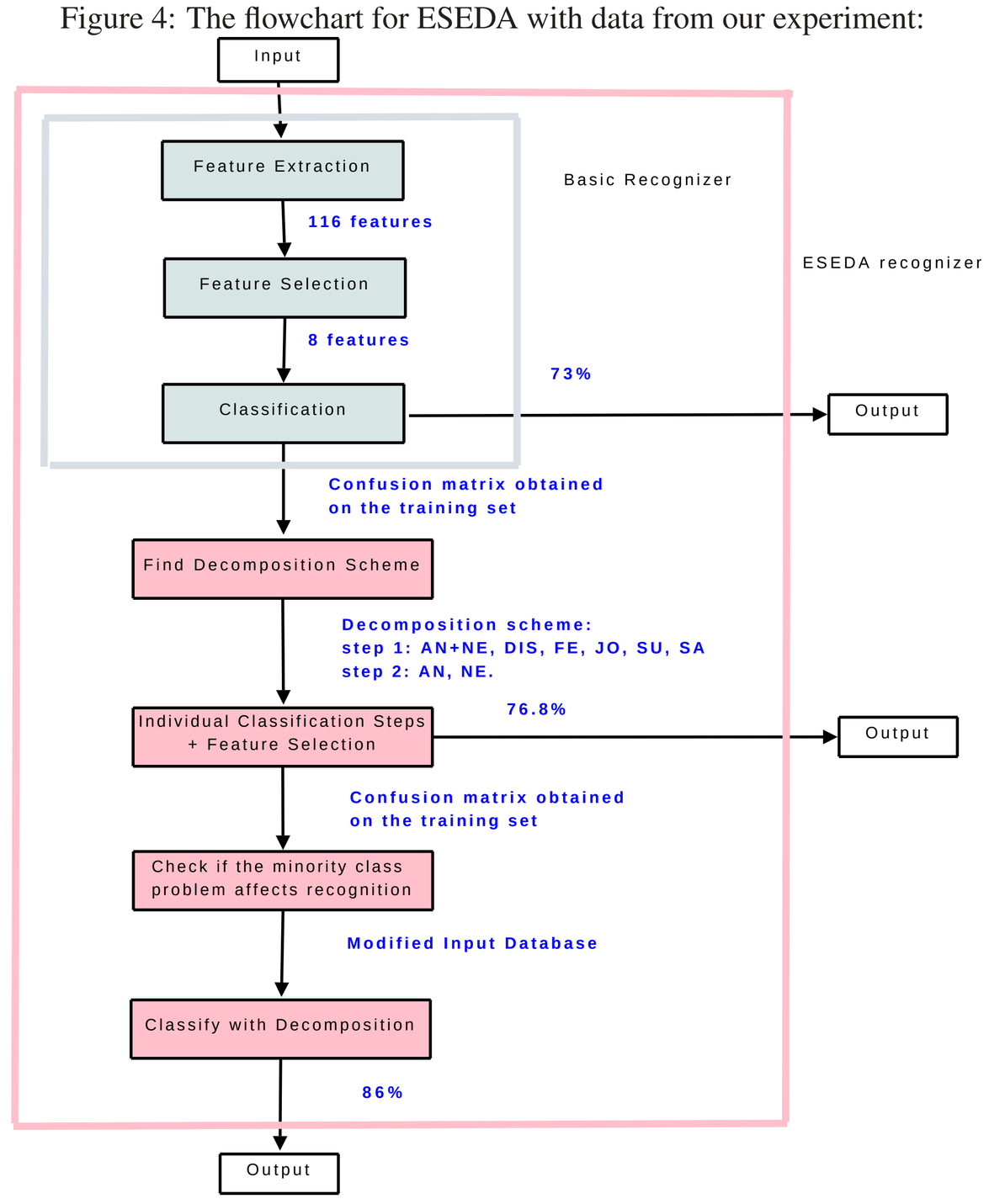
3 Treatment of the Minority Class Problem

- If some class is not well recognised ($\leq 70\%$), check if this is a minority class (≤ 500 samples in the training set);
- If it is a minority class, duplicate each sample of this class in the training set.

Emotion	Accuracy	Number of samples	MCP treatment?
Neutral	76%	389 samples	No
Anger	70%	313 samples	Yes
Disgust	94%	705 samples	No
Fear	53%	700 samples	No
Joy	83%	689 samples	No
Surprise	63%	525 samples	No
Sad	72%	700 samples	No

4 Flowchart for ESEDA

Figure 4: The flowchart for ESEDA with data from our experiment:



5 Results and Conclusions

	Anger	Neutral	Total
Baseline	70%	76%	73.3%
+ classification decomposition	84%	95%	76.8%
+ MCP treatment	99.5%	93%	86%

Our optimisations are simple from a theoretical point of view, yet lead to good accuracy improvements. We will analyse how sensible the approach is to speaker, language etc.

6 Acknowledgements

This research was supported by AGAUR, the Research Agency of Catalonia, with the BE-DRG 2007 grant.

التكنولوجيا اللغوية ومساعدتها فى تطوير اللغة العربية للصم وضعاف السمع

Safinaz HUSEEN GODA

الأصم من الناحية التربوية

- هو الشخص الذى لا يستطيع الأعتماد على حاسة السمع لتعلم اللغة أو الأستفادة من برامج التعليم المختلفة . ويكون هذا الشخص بحاجة الى برامج تعليمية مختلفة خاصة تعوضه عن فقدان السمع .

وظائف اللغة للأصم وأشكال الأتصال

1* التواصل بين الناس وتبادل المعرفة والمشاعر وأرساء دعائم التفاهم والحياة المشتركة .

2* التعبير عن حاجة الفرد المختلفة.

3* النمو الذهنى المرتبط بالنمو اللغوى وتعلم اللغة الشفوية أو الأشارية يولد لدى الفرد المفاهيم والصور الذهنية .

4* أرتباط اللغة بأطار حضارية متعمق مع التاريخ والمجتمع.

5* الوظيفة النفسية فاللغة تنفس عن الإنسان وتخفف من حدة الضغوط الداخلية ويبدو ذالك فى مواقف الأنفعال.

تعتمد أنظمة الأتصال لدى الأصم على الأتصال الشفوى أو الأتصال الأشارى من خلال:-

1* الأسلوب الشفوى أو قراءة الشفاه.

2* الاشارات اليدوية المساعدة لتعليم النطق .

3* لغة التلميح.

4* أبجدية الأصابع الاشارية أو الهجاء بالأصابع.

5* طريقة اللفظ المنغم.

6* لغة الاشارة .

7* الاتصال الشامل (الكلى) .

ماهو تاريخ لغة الأشارة؟

- ترجع أقدم المحاولات المعروفة المتصلة بتنمية قدرات الاتصال لدى الصم الى رجلى دين فى الكنيسة الكاثلوكية وقد عاشا فى القرن 17 :

الأول أسبانى : -

بدروبانس دوليون: أهتم بتنمية التواصل الشفوى لدى الصم ونجح فى تعليم قراءة اللغة الاتينية لشقيقين أصمين وطريقة لا تبعد عن الطريقة الشفوية الحالية المعتمدة على قراءة الشفاه.

الثانى فرنسى :-

دولابى : كانت تتسم دائما بالمحلية فتختلف من بلد الى أخرى ومن جهة أخرى وأول من بادر الى تنظيمها وتقنينها هو الأب دولابى الذى نظم الاشارة التى يستعملها الصم ودونها فى قاموس صغير وأصبحت هذة اللغة الأساسية فى المدارس التى كان يشرف عليها .

من المساهمين فى نشر هذة اللغة

غالودية . الذى سافر سنة 1817 الى أمريكا وأسس مدرسة لتعليم الصم تحمل الى اليوم أسمه بعد ما تطورا الى ان أصبحت اليوم اول جامعة فى العالم تعتنى بالتعليم العالى للصم والبحوث والدراسات ويرأسها عميد أصم ويشكل الصم نسبة عالية من الأساتذة وتعتمد فيها لغة الأشارة فى الدرجة الأولى.

ظهر أهتمام فى الدول الأسكندنافية بها أما أوربا الغربية ولا سيما فرنسا وأيطاليا وبلجيكا وأسبانيا فلم تهتم بهذة اللغة الا فى منتصف السبعينيات .

أدوات التكنولوجيا الحديثة المساعدة لتطوير التواصل للصم

- الكمبيوتر وشبكات النت المختلفة .
- التليفون المحمول المزود ب(2) كاميرا .
- الأجهزة الطبية المختلفة لمساعدة الأصم وتدريبهم لمخارج الحروف الأبجدية على الجهاز الكلامى وأيضا تجويد نطق الحرف.
- برامج الأخبار وترجمتها للأشارة فى التلفزيون وقنوات وبرامج دينية مختلفة .

ملحوظة :-

الأصم وضعاف السمع من خصائصة وصفاتة هى :-(النسيان) للأحداث والمواقف .

فمن تجربتى الخاصة والتعامل معهم وجدت وسيلة سهلة لتوصيل المعلومة من الجانب الدينى فقط وهى القصص بالصلصال حيث يشاهد الحدث بتسلسل + الى جانب الترجمة باللغة العربية المبسطة وفى خلال 9 شهور كانت النتيجة إيجابية جدا وممتازة ويمكن تطبيقها على أى جانب لتطوير اللغة لديهم .

الفكرة المقترحة

- القاموس الإشارى الموحد المصور بالفيديو وأيضا مدعم بالكلمات وتركيب الجمل البسيطة على الموبيل و cd كمبيوتر الى جانب برامج تعليمية أخرى.

لاختيار الوسائل التعليمية مراعاة أن :-

1- ترتبط بموضوع الدرس بشكل مباشر.
2- تساعد فى تحقيق أهداف الدرس.
3-تناسب مستوى النمو للطفل وخبراته السابقة .
4-تكون جذابة وشيقة ولا تمثل خطورة بالنسبة للطفل .
5-تكون بسيطة وغير معقدة حتى لا تعرض الطفل للارتباك والتشتت والإحباط .
6-ترتبط ببيئة الطفل المحلية وتساعده فى اكتساب المهارات اللازمة لحياته اليومية فى المجتمع .
7- متنوعة ومبتكرة بحيث لا تعرض الطفل للضيق والملل.
8-ومن الوسائل التى يمكن استخدامها مع هؤلاء الأطفال :-
الصور - والشفافيات – والنماذج التعليمية – والأفلام والخرائط – ومجلات الحائط –والزجاجات البلاستيكية – والأكواب الفارغة – وعلب البيبسى – والألعاب البلاستيكية والحروف المغناطيسية.

Statistical Language Modeling using SRILM Toolkit

Presented by:
Kamal Eldin Mahmoud

AGENDA

- **Introduction**
- **Basic SRILM Tools**
  - **ngram-count**
  - **ngram**
  - **ngram-merge**
- **Basic SRILM file format**
  - **ngram-format**
  - **nbest-format**

AGENDA

Basic SRILM Scripts

- **Training-scripts**
- **lm-scripts**
- **ppl-scripts**

Introduction

- SRILM is a collection of C++ libraries, executable programs, and helper scripts.

- The toolkit supports creation and evaluation of a variety of language model types based on N-gram statistics.

- The main purpose of SRILM is to support language model estimation and evaluation.

- Since most LMs in SRILM are based on N-gram statistics, the tools to accomplish these two purposes are named ngram-count and ngram, respectively.

Introduction

- A standard LM (trigram with Good-Turing discounting and Katz backoff for smoothing) would be created by

ngram-count -text TRAINDATA -lm LM

- The resulting LM may then be evaluated on a test corpus using

ngram -lm LM -ppl TESTDATA -debug 0

Basic SRILM Tools

ngram-count

ngram-count generates and manipulates N-gram counts, and estimates N-gram language models from them.

Syntax:

Ngram-count *[* ***-help*** *]* *option ...*

ngram-count options

Each filename argument can be an ASCII file, or a compressed file (name ending in .Z or .gz)

-help
Print option summary.
-version
Print version information.
-order n
Set the maximal order (length) of N-grams to count. This also determines the order of the estimated LM, if any. The default order is 3.
-memuse
Print memory usage statistics.

ngram-count options

-vocab *file*
Read a vocabulary from file.

-vocab-aliases *file*
Reads vocabulary alias definitions from file, consisting of lines of the form

alias word

This causes all tokens alias to be mapped to word.

-write-vocab *file*
-write-vocab-index *file*
Write the vocabulary built in the counting process to file.

ngram-count counting options

-tolower
Map all vocabulary to lowercase.

-text *textfile*
Generate N-gram counts from text file.

-no-sos
Disable the automatic insertion of start-of-sentence tokens in N-gram counting.
-no-eos
Disable the automatic insertion of end-of-sentence tokens in N-gram counting.

-read *countsfile*
Read N-gram counts from a file.

ngram-count counting options

-**read-google** *dir*
Read N-grams counts from an indexed directory structure rooted in dir, in a format developed by Google. The corresponding directory structure can be created using the script *make-google-ngrams* .

-write *file*
-**write-binary** *file*
-write-order n
-writen *file*
Write total counts to file.

-sort
Output counts in lexicographic order, as required for ngram-merge.

ngram-count lm options

-lm *lmfile*
-write-binary-lm
Estimate a backoff N-gram model from the total counts, and write it to *lmfile* .

-unk
Build an ``open vocabulary'' LM.

-map-unk *word*
Map out-of-vocabulary words to *word*.

ngram-count lm options

-cdiscount*n* *discount*
Use Ney's absolute discounting for N-grams of order *n*, using *discount* as the constant to subtract.

-wbdiscount*n*
Use Witten-Bell discounting for N-grams of order *n*.

-ndiscount*n*
Use Ristad's natural discounting law for N-grams of order *n*.

-addsmooth*n* *delta*
Smooth by adding *delta* to each N-gram count.

ngram-count lm options

-kndiscount*n*
Use Chen and Goodman's modified Kneser-Ney discounting for N-grams of order *n*.

-kn-counts-modified
Indicates that input counts have already been modified for Kneser-Ney smoothing.

-interpolate*n*
Causes the discounted N-gram probability estimates at the specified order *n* to be interpolated with lower-order estimates. Only Witten-Bell, absolute discounting, and (original or modified) Kneser-Ney smoothing currently support interpolation.

ngram

Ngram performs various operations with N-gram-based and related language models, including sentence scoring, and perplexity computation.

Syntax:

ngram *[* ***-help*** *] option …*

ngram options

-help
Print option summary.

-version
Print version information.

-order n
Set the maximal N-gram order to be used, by default 3.

-memuse
Print memory usage statistics for the LM.

ngram options

The following options determine the type of LM to be used.

-null

Use a `null' LM as the main model (one that gives probability 1 to all words).

-use-server *S*

Use a network LM server as the main model.

-lm *file*

Read the (main) N-gram model from *file*.

ngram options

-tagged
Interpret the LM as containing word/tag N-grams.

-skip
Interpret the LM as a ``skip'' N-gram model.

-classes *file*
Interpret the LM as an N-gram over word classes.

-factored
Use a factored N-gram model.

-unk
Indicates that the LM is an open-class LM.

ngram options

-ppl *textfile*
Compute sentence scores (log probabilities) and perplexities from the sentences in *textfile*.
The **-debug** option controls the level of detail printed.

-debug 0
Only summary statistics for the entire corpus are printed.

-debug 1
Statistics for individual sentences are printed.

ngram options

-debug 2
Probabilities for each word, plus LM-dependent details about backoff used etc., are printed.

-debug 3
Probabilities for all words are summed in each context, and the sum is printed.

ngram options

-nbest *file*
Read an N-best list in nbest-format and rerank the hypotheses using the specified LM. The reordered N-best list is written to stdout.

-nbest-files *filelist*
Process multiple N-best lists whose filenames are listed in *filelist*.

-write-nbest-dir *dir*
Deposit rescored N-best lists into directory *dir*, using filenames derived from the input ones.

ngram options

-decipher-nbest
Output rescored N-best lists in Decipher 1.0 format, rather than SRILM format.

-no-reorder
Output rescored N-best lists without sorting the hypotheses by their new combined scores.

-max-nbest *n*
Limits the number of hypotheses read from an N-best list.

ngram options

-no-sos
Disable the automatic insertion of start-of-sentence tokens for sentence probability computation.

-no-eos
Disable the automatic insertion of end-of-sentence tokens for sentence probability computation.

ngram-merge

ngram-merge reads two or more lexicographically sorted N-gram count files and outputs the merged, sorted counts.

Syntax:

ngram-merge [-help] [-write outfile] [-float-counts] \ [--] infile1 infile2 ...

Ngram-merge options

-write *outfile*
Write merged counts to *outfile*.

-float-counts
Process counts as floating point numbers.

--
Indicates the end of options, in case the first input filename begins with ``-".

Basic SRILM file format

ngram-format

ngram-format File format for ARPA backoff N-gram models

\data
ngram 1=*n1*
ngram 2=*n2*.
..
ngram *N*=*nN*
\1-grams:
p *w* [*bow*]
...****
2-grams:
p *w1 w2* [*bow*]
...
*****N***-grams:***
p *w1 ... wN*
...
\end

nbest-format

SRILM currently understands three different formats for lists of N-best hypotheses for rescoring or 1-best hypothesis extraction. The first two formats originated in the SRI Decipher(TM) recognition system, the third format is particular to SRILM.

The first format consists of the header

NBestList1.0

followed by one or more lines of the form

(*score*) *w1 w2 w3* ...

where *score* is a composite acoustic/language model score from the recognizer, on the bytelog scale.

nbest-format

The second Decipher(TM) format is an extension of the first format that encodes word-level scores and time alignments. It is marked by a header of the form

NBestList2.0

The hypotheses are in the format

(*score*) *w1* (st: *st1* et: *et1* g: *g1* a: *a1*) *w2* ...

where words are followed by start and end times, language model and acoustic scores (bytelog-scaled), respectively.

nbest-format

The third format understood by SRILM lists hypotheses in the format

ascore lscore nwords w1 w2 w3 ...

where the first three columns contain the acoustic model log probability, the language model log probability, and the number of words in the hypothesis string, respectively. All scores are logarithms base 10.

Basic SRILM Scripts

Training-scripts

These scripts perform convenience tasks associated with the training of language models.

get-gt-counts

Syntax

get-gt-counts max=*K* **out=***name* [*counts* ...] **>** *gtcounts*

Computes the counts-of-counts statistics needed in Good-Turing smoothing. The frequencies of counts up to *K* are computed (default is 10). The results are stored in a series of files with root *name*, ***name*.gt1counts**,..., ***name*.gt*N*counts**.

Training-scripts

make-gt-discounts

Santax:

make-gt-discounts min=*min* max=*max gtcounts*

Takes one of the output files of get-gt-counts and computes the corresponding Good-Turing discounting factors. The output can then be passed to **ngram-count** via the **-gt*n*** options to control the smoothing during model estimation.

Training-scripts

make-abs-discount

Syntax

make-abs-discount *gtcounts*

Computes the absolute discounting constant needed for the **ngram-count -cdiscount***n* options. Input is one of the files produced by **get-gt-counts**.

Training-scripts

make-kn-discount

Syntax

make-kn-discounts min=*min gtcounts*

Computes the discounting constants used by the modified Kneser-Ney smoothing method. Input is one of the files produced by **get-gt-counts**.

Training-scripts

make-batch-counts

Syntax

make-batch-counts *file-list* \ [*batch-size* [*filter* [*count-dir* [*options* ...]]]]

Performs the first stage in the construction of very large N-gram count files. *file-list* is a list of input text files. Lines starting with a `#' character are ignored. These files will be grouped into batches of size *batch-size* (default 10). The N-gram count files are left in directory *count-dir* (``counts'' by default), where they can be found by a subsequent run of **merge-batch-counts**.

Training-scripts

merge-batch-counts

Syntax

merge-batch-counts *count-dir* [*file-list*|*start-iter*]

Completes the construction of large count files. Optionally, a *file-list* of count files to combine can be specified. A number as second argument restarts the merging process at iteration *start-iter*.

Training-scripts

make-google-ngrams

Syntax

make-google-ngrams [**dir=***DIR*] [**per_file=***N*] [**gzip=0**] \ [**yahoo=1**] [*counts-file* ...]

Takes a sorted count file as input and creates an indexed directory structure, in a format developed by Google to store very large N-gram collections. Optional arguments specify the output directory *dir* and the size *N* of individual N-gram files (default is 10 million N-grams per file). The **gzip=0** option writes plain. The **yahoo=1** option may be used to read N-gram count files in Yahoo-GALE format.

Training-scripts

tolower-ngram-counts

Syntax

tolower-ngram-counts [*counts-file* ...]

Maps an N-gram counts file to all-lowercase. No merging of N-grams that become identical in the process is done.

Training-scripts

reverse-ngram-counts

Syntax

reverse-ngram-counts [*counts-file* ...]

Reverses the word order of N-grams in a counts file or stream.

reverse-text

Syntax

reverse-text [*textfile* ...]

Reverses the word order in text files, line-by-line.

Training-scripts

compute-oov-rate

Syntax

compute-oov-rate *vocab* [*counts* ...]

Determines the out-of-vocabulary rate of a corpus from its unigram *counts* and a target vocabulary list in *vocab*.

lm-scripts

add-dummy-bows

Syntax

add-dummy-bows [*lm-file*] **>** *new-lm-file*

Adds dummy backoff weights to N-grams, even where they are not required, to satisfy some broken software that expects backoff weights on all N-grams (except those of highest order).

lm-scripts

change-lm-vocab

Syntax

change-lm-vocab -vocab *vocab* **-lm** *lm-file* **-write-lm** *new-lm-file* \ [**-tolower**] [**-subset**] [*ngram-options* ...]

Modifies the vocabulary of an LM to be that in *vocab*. Any N-grams containing OOV words are removed, new words receive a unigram probability, and the model is renormalized. The **-tolower** option causes case distinctions to be ignored. **-subset** only removes words from the LM vocabulary, without adding any.

lm-scripts

make-lm-subset

Syntax

make-lm-subset *count-file*|**-** [lm-file |**-**] **>** *new-lm-file*

Forms a new LM containing only the N-grams found in the *count-file*. The result still needs to be renormalized with **ngram -renorm** .

lm-scripts

get-unigram-probs

Syntax

get-unigram-probs [**linear=1**] [*lm-file*]

Extracts the unigram probabilities in a simple table format from a backoff language model. The **linear=1** option causes probabilities to be output on a linear (instead of log) scale.

ppl-scripts

These scripts process the output of the ngram option **-ppl** to extract various useful information.

add-ppls

Syntax

add-ppls [*ppl-file* ...]

Takes several ppl output files and computes an aggregate perplexity and corpus statistics.

ppl-scripts

subtract-ppls

Syntax

subtract-ppls *ppl-file1* [*ppl-file2* ...]

Similarly computes an aggregate perplexity by removing the statistics of zero or more *ppl-file2* from those in *ppl-file1*.

ppl-scripts

compare-ppls

Syntax

compare-ppls [**mindelta=***D*] *ppl-file1 ppl-file2*

Tallies the number of words for which two language models produce the same, higher, or lower probabilities. The input files should be **ngram -debug 2 -ppl** output for the two models on the same test set. The parameter *D* is the minimum absolute difference for two log probabilities to be considered different.

ppl-scripts

compute-best-mix

Syntax

compute-best-mix [**lambda='***l1 l2* ...**'**]
[**precision=***P*] \ *ppl-file1* [*ppl-file2* ...]

Takes the output of several **ngram -debug 2 –ppl** runs on the same test set and computes the optimal interpolation weights for the corresponding models. Initial weights may be specified as *l1 l2 ...*. The computation is iterative and stops when the interpolation weights change by less than *P* (default 0.001).

ppl-scripts

compute-best-sentence-mix

Syntax

compute-best-sentence-mix [**lambda='***l1 l2* ...**'**] [**precision=***P*] \ *ppl-file1* [*ppl-file2* ...]

similarly optimizes the weights for sentence-level interpolation of LMs. It requires input files generated by **ngram -debug 1 -ppl**.

Thank You ☺