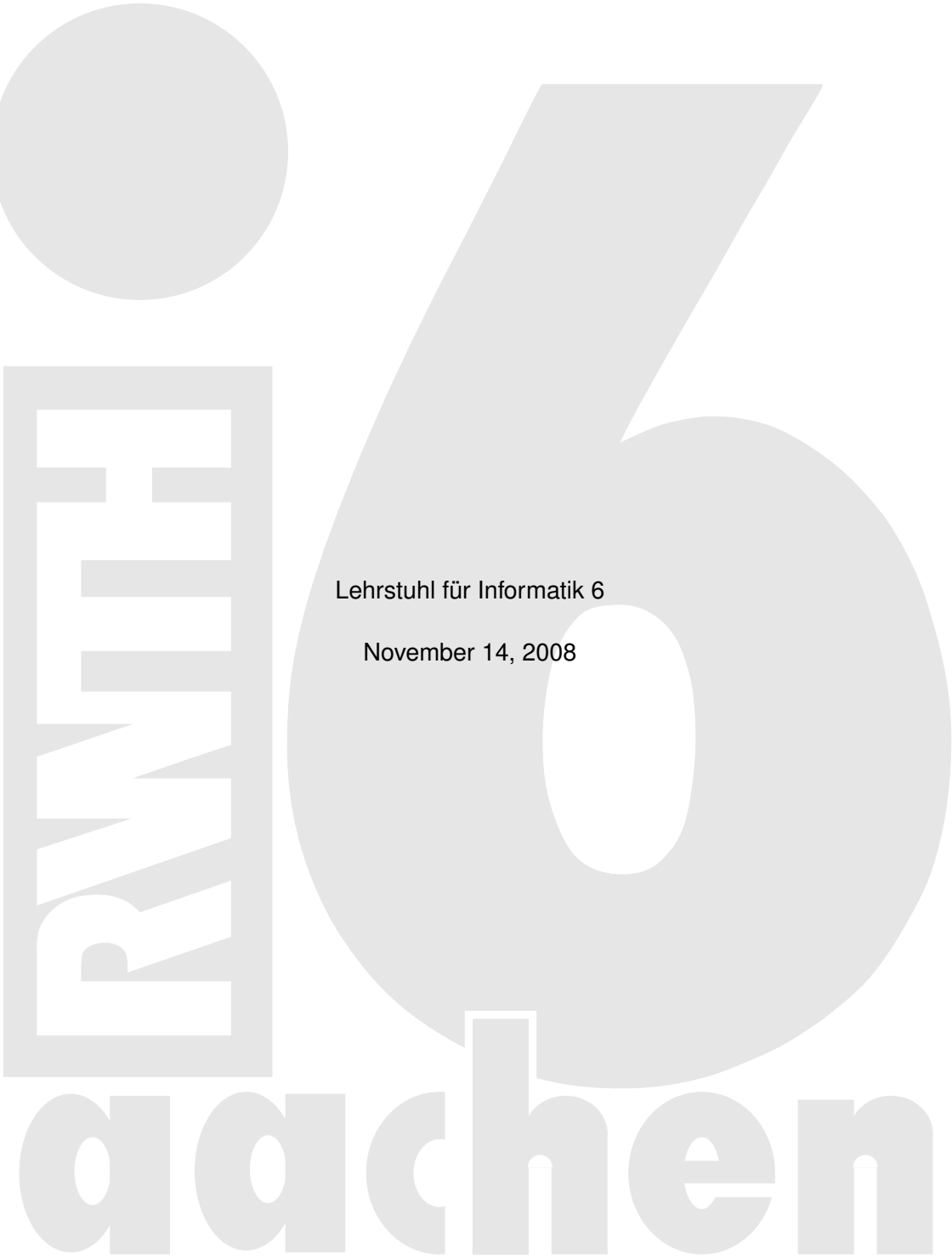
Bachelor Proposals



Lehrstuhl für Informatik 6

November 14, 2008

CONTENTS

1.1 Face Recognition using Distortion Models

WHAT:

- implement a nearest neighbor based face recognizer with distance measures that can handle face and image variations
- evaluate the proposed framework for different appearance-based features and distance measures on state-of-the-art databases (e.g. NIST FRGC, AR-Face, and/or CMU-PIE)

WHY:

- invariance against illumination, occlusions, and poses are challenging problems in face recognition
- most approaches are based on an eye-detection based alignment of the face as a preprocessing step
- recent research results have shown that the alignment is not necessarry, and that 2D-DCT based frequency features perform well under different illuminations

HOW:

- extend existing feature extaction framework: gabor and 2D-DCT frequency based features
- we propose to use image distortion models successfully used in OCR to improve the face recognition

SCHEDULE:

1. Month
    - check literature, especially papers from UKA about feature extraction and from RWTH about distortion models
        - H.K. Ekenel, R. Stiefelhagen, Block Selection in the Local Appearance-based Face Recognition Scheme, CVPR Biometrics Workshop, New York, USA, June 2006.
        - H.K. Ekenel, R. Stiefelhagen, Analysis of Local Appearance-based Face Recognition: Effects of Feature Selection and Feature Normalization, CVPR Biometrics Workshop, New York, USA, June 2006.

  - D. Keysers, T. Deselaers, C. Gollan, and H. Ney. Deformation Models for Image Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 29, number 8, pages 1422-1435, August 2007.
- checkout and compile w2d distortion library
- download/prepare evaluation databases
- run baseline experiments using intensity features and Euclidean Distance

2. Month
   - implement/extend feature extraction framework (either external in Matlab or in w2d/C/C++)
   - create ascii feature files (Jörg Files) to be used within the w2d libray
3. Month
   - evaluate influence of different features and distortion models
   - start writing
4. Month
   - analyze results (visualizations, error plots, ...)
   - writing (about 50 pages)

CONTACT:

- Philippe Dreuw `dreuw@cs.rwth-aachen.de`
- Thomas Deselaers `deselaers@cs.rwth-aachen.de`

1.2 Diversity-oriented Image Retrieval

WHAT:

- extend our current image retrieval system to deliver diverse results
- hardly anything has been done in this area
- only thing I know:
  - ImageCLEF 2008 photo retrieval task

HOW:

- extend FIRE to have a clustering step after retrieval (should be doable in one class... aka. e.g. RetrievalResultResorter

SCHEDULE:

1. Month
   - get to know FIRE
   - baseline experiments on ImageCLEF Photo Retrieval Task
   - simple clustering as postprocessing step
   - incorporation of text information
   - maybe: think about a second database for evaluation (e.g. get some google/flickr queries, EXALEAD database, ...)
2. Month
   - implement RetrievalResultResorter
   - perform first serious experiments
3. Month
   - implement different Clustering techniques for resorting
   - evaluate these
   - start writing
4. Month
   - writing
   - fill in gaps in experiments section
   - comparison with stuff from ImageCLEF photo

CONTACT:

- Thomas Deselaers `deselaers@cs.rwth-aachen.de`
- Philippe Dreuw `dreuw@cs.rwth-aachen.de`

1.3 Arabic Handwriting Recognition with OCRopus and RWTH-OCR

WHAT:

- extend the open-source software OCRopus in order to recognize cursive handwritings
  - OCRopus is partly based on `http://pubs.iupr.org/#1995-breuel-das`
  - `http://groups.google.com/group/ocropus/browse_thread/thread/66c9741f94d65b5e`
  - `http://ipet.iupr.org/demos.html#DIVER`
- only use preprocessing framework of OCRopus in combination with RWTH-OCR as line recognizer
- Arabic Handwriting Recognition

HOW:

- open-source OCRopus `http://sites.google.com/site/ocropus/`
- use LUA script language to create interface to RWTH-ASR / RWTH-OCR

SCHEDULE:

1. Month
   - get to know OCRopus (download, install, examples, ...)
   - perpare Arabic IFN/ENIT handwriting database (and/or French RIMES, English IAM database)
   - setup OCRopus to extract lines
2. Month
   - run baseline (offline) experiment with RWTH-OCR on cursive handwriting database and OCRopus extracted lines
   - extend OCRopus and write a RWTH-OCR interface in order to recognize cursive handwriting
3. Month
   - run experiments using OCRopus/RWTH-OCR, analyze preprocessing tools
   - start writing
4. Month
   - compare to RWTH-OCR results w/o preprocessing
   - writing

CONTACT:

- Philippe Dreuw dreuw@cs.rwth-aachen.de

2.1 Log-linear modeling using advanced features

WHAT:

- state-of-the-art acoustic modeling: Gaussian hidden Markov models
- use log-linear models to refine conventional acoustic models
- by augmenting the existing features
- evaluate these features on different speech recognition tasks

WHY:

- all information and non-linearities come in through the features
- log-linear models are a flexible framework to add features
- convex optimization of log-linear models (no spurious local optima)

HOW:

- example features: detector-based features (e.g. posterior or neural network features), n-th order features etc.
- efficiency is an essential issue
- tasks: running (large-scale) experiments, scripting, programming

SCHEDULE:

1. Month
   - get familiar with log-linear models in speech recognition
     - E. Fosler-Lussier and J. Morris, "CRANDEM systems: Conditional Random Field Acoustic Models for Hidden Markov Models," Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2008), Las Vegas, Nevada, 2008.
     - H.-K. J. Kuo and Y. Gao, "Maximum entropy direct models for speech recognition," IEEE Transactions on Audio, Speech, and Language Processing, volume 14, number 3, pages 873-881, May 2006.
     - J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in Proc. of the 18th Int. Conf. on Machine Learning, 2001.
   - prepare experimental setup and produce first baseline results

- check out internal software tools

2. Month
   - implement/extend feature extraction framework in software tool
   - test augmented features on a small task
3. Month
   - evaluate augmented features for a large vocabulary continuous speech recognition task
   - analyze results and refine/add features
4. Month
   - analyze results
   - documentation

CONTACT:

- Georg Heigold `heigold@cs.rwth-aachen.de`
- Ralf Schlüter `schlueter@cs.rwth-aachen.de`

2.2 Refined Discriminative Training Criteria

WHAT:

- estimation of the acoustic model parameters is essential
- discriminative training criteria are established in speech recognition
- test refined error-based training criteria that are defined on state level

WHY:

- existing training criteria are only an approximation to the ideal test error
- can we find a better approximation? (in terms of final error rate)

HOW:

- evaluate this criteria on state-of-the-art LVCSR tasks
- comparison with conventional training criteria

SCHEDULE:

1. Month
   - become acquainted with discriminative training in speech recognition
     - R. Schlüter, "Investigations on Discriminative Training Criteria," PhD Thesis, Aachen, Germany, September 2000.
     - G. Heigold, T. Deselaers, R. Schlüter, and H. Ney, "Modified MMI/MPE: A Direct Evaluation of the Margin in Speech Recognition," in Int. Conf. on Machine Learning, pages 384-391, Helsinki, Finland, July 2008.
   - this also includes our software tool used for discriminative training
   - run first discriminative training to get familiar with experimental setup
2. Month
   - modify software/configuration to include the new training criteria
   - run first experiments with this criteria
3. Month
   - tune training criteria/experimental setup
   - analyze experimental results
4. Month
   - documentation

CONTACT:

- Georg Heigold heigold@cs.rwth-aachen.de
- Ralf Schlüter schlueter@cs.rwth-aachen.de

2.3 Italian Speech Recognition of Court Cases

WHAT:

- Use the RWTH automatic speech recognition system (RWTH ASR) to build , train, and evaluate, an Italian speech recognition system.

HOW:

- Define pronunciation lexicon, and train language model for Italian
- Use our existing Spanish speech recognition system to perform first recognition
- Retrain and refine acoustic model using unsupervised training

SCHEDULE:

1. Month
   - Get familiar with speech recognition using RWTH ASR.
   - Choose and prepare Italian acoustic evaluation data
   - Get familiar with language models for speech recognition
   - Collect and choose italian language model data
   - Train Italian language model
2. Month
   - Get familiar with phonetic pronunciation lexica for speech recognition
   - Develop an Italian pronunciation lexicon
     - Prepare basic lexicon
     - Train grapheme-to-phoneme models
     - Prepare pronounciation lexicon matching the language model
   - Construct a mapping between Italian and Spanish phonemes
   - Perform first Italian speech recognition:
     - Using Spanish ASR system
     - Language model, lexicon, and Spanish-Italian phoneme mapping as created above
3. Month
   - Start writing
   - Choose and prepare Italian acoustic model training data
   - Perform recognition of untranscribed acoustic training data

- Perform a first iteration acoustic model training on untranscribed (and optionally transcribed) acoustic training data
- Evaluate trained Acoustic model

4. Month
    - Continue writing
    - Iterative refinement of acoustic model
    - Evaluate refinements to acoustic model

CONTACT:

- Jonas Lööf `loof@cs.rwth-aachen.de`
- Ralf Schlüter `schluter@cs.rwth-aachen.de`

2.4 Optimization of Search Algorithms

WHAT:

- analyze and optimize the word conditioned tree search used by our speech recognizer
- evaluate different pruning methods
- implement and evaluate different types of search space organization

WHY:

- optimization of the search is one of the key points to increase the speed of the recognition process

HOW:

- optimize parameters of existing pruning methods
- implement a search space organization with different types of trees

SCHEDULE:

1. Month
   - read literature about the used search algorithm, pruning methods, and search space organization
   - get to know the existing implementation
   - prepare a setup for experiments
2. Month
   - run experiments using different pruning parameters and pruning methods
   - implement search space organization using extended word pair approximation
3. Month
   - run experiments comparing both types of search space organization
   - start writing
4. Month
   - run remaining experiments
   - writing

CONTACT:

- David Rybach `rybach@cs.rwth-aachen.de`

3.1 Efficient Training for Machine Translation Models

WHAT:

- Implement an efficient training corpus representation which in practice suppresses the need of a (time and space intensive) separate training process
- Evaluate the approach on a hierarchical phrase machine translation framework by exhaustively testing different training parameters.

WHY:

- The training of statistical translation models is a time consuming process, which limits the capacity of exhaustively testing different phrase extraction parameters. This is specially true for syntax-based or hierarchical machine translation models.
- Furthermore the large amount of possible phrases makes necessary techniques like filtering on the test corpus, which limits the usefulness of a trained system.

HOW:

- Extend the existing software in the institute by adding a new training component
- Evaluate the approach on standard databases used in international evaluation campaigns.

SCHEDULE:

1. Month
   - check literature, especially
     - Adam Lopez, Tera-Scale Translation Models via Pattern Matching. In Proceedings of COLING, pp. 505-512, August 2008.
     - Adam Lopez, Machine Translation by Pattern Matching. Dissertation, University of Maryland, March 2008.
   - Familiarize yourself with the software framework at the institute
2. Month
   - Implement the efficient training corpus representation
3. Month
   - Adapt the translation process in order to use the efficient representation
   - Conduct translation experiments

- Start writing

4. Month
    - Analyze results
    - Writing (about 50 pages)

CONTACT:

- David Vilar `vilar@cs.rwth-aachen.de`

3.2 A Paraphrasing MT System

WHAT:

- In this work, a paraphrasing system is to be build as a English-to-English-Machine translation system.
- Evaluate the approach on a phrase-based machine translation framework.

WHY:

- When evaluating the output of Machine Translation systems, reference translations play an important role. Having multiple alternative reference translations for a single source sentence can be very helpful to tackle problems with ambiguities, but these are costly to acquire, and in many cases, not available.
- It has been reported that automatically generated paraphrases of the reference translations can be helpful to get more consistent results when tuning a Machine Translation system.

HOW:

- As there is hardly any ”bilingual” E-E training data available, phrase tables, word lexica, etc should be learned from (provided) bilingual tables, lexica etc., e.g. Chinese-English
- Evaluate the approach on standard databases used in international evaluation campaigns.

SCHEDULE:

1. Month
   - check literature, especially
     - Chris Callison-Burch, 2008. Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In Proceedings of EMNLP 2008.
     - Chris Callison-Burch, Philipp Koehn and Miles Osborne, 2006. Improved Statistical Machine Translation Using Paraphrases. In Proceedings NAACL-2006.
   - Familiarize yourself with the software framework at the institute
2. Month
   - implement different approaches to generate English-to-English phrase tables, lexica, etc. from bilingual phrase tables provided by i6
3. Month

- run optimization experiments for the remaining parameters
- if time permits, different sources/languages for the bilingual ”training” tables can be tried, and/or the effect of having paraphrased references available for MT system tuning
- Start writing

4. Month
    - Analyze results
    - Writing (about 50 pages)

CONTACT:

- Gregor Leusch `leusch@cs.rwth-aachen.de`

3.3 Multilingual Instant-Messaging

WHAT:

- Development of a instant messaging proxy that translarently translatate chat messages between languages
- Interfacing with existing translation server technology
- Training of an machine translation system suitable for translating chat messages for one language pair

HOW:

- implementation of instant messaging proxy/plugin
- prepare corpus, train translation system
- create working showcase

SCHEDULE:

1. Month
   - check for existing open source solutions
   - implement new or adapt existing software for the task
   - testing of interface
2. Month
   - adapt preprocessing and select corpora
   - train machine translation system
   - run baseline experiments
3. Month
   - test and improve MT pipeline
   - start writing
4. Month
   - writing
   - fill in gaps in experiments section
   - final demonstration setup

CONTACT:

- Arne Mauser `mauser@cs.rwth-aachen.de`

3.4 Distributed Language Models

WHAT:

- implement a distributed language model server that is capable of delivering n-gram probabilities with a high throughput (using TCP/IP)
- evaluate the speed and robustness of the server on real translation tasks

WHY:

- one weakness of large-scale MT systems is the size and loading time of the language model
- a distributed LM server allows for small memory footprints of the decoder
- more flexibility for switching genre-specific LMs on-the-fly

HOW:

- enhance an existing LM toolkit with distributed LM lookup capabilities
- write an interface that can be used in the decoder to easily access n-gram lookup

SCHEDULE:

1. Month
   - introductory literature:
     - T. Brants, A. Popat, P. Xu, F. J. Och, J. Dean. "Large Language Models in Machine Translation", Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing, 2007.
     - A. Emami, K. Papineni, J. Sorensen. "Large-Scale Distributed Language Modeling", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2007.
     - Y. Zhang, A. S. Hildebrand, S. Vogel. "Distributed Language Modeling for N-best List Re-ranking", Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, 2006.
   - get familiar with SRILM toolkit and i6 TCP/IP server add-ons (used for distributed rescoring)
2. Month
   - implement/extend distributed LM server framework (in SRILM using C++)
   - get familiar with our phrase-based decoder by testing different language models (translation experiments)
3. Month

- implement interface that can be used within our phrase-based decoder to access LM server functionality
- stress tests, gather statistics on speed and robustness

4. Month
    - analyze results
    - write thesis (about 50 pages)

CONTACT:

- Saša Hasan hasan@cs.rwth-aachen.de