Machine Learning Tutorial

CB, GS, REC

Section 5a
UIMA + ML = ClearTK-ML

Machine Learning Tutorial for the UKP lab
June 10, 2011

What to learn?

TECHNISCHE UNIVERSITÄT DARMSTADT

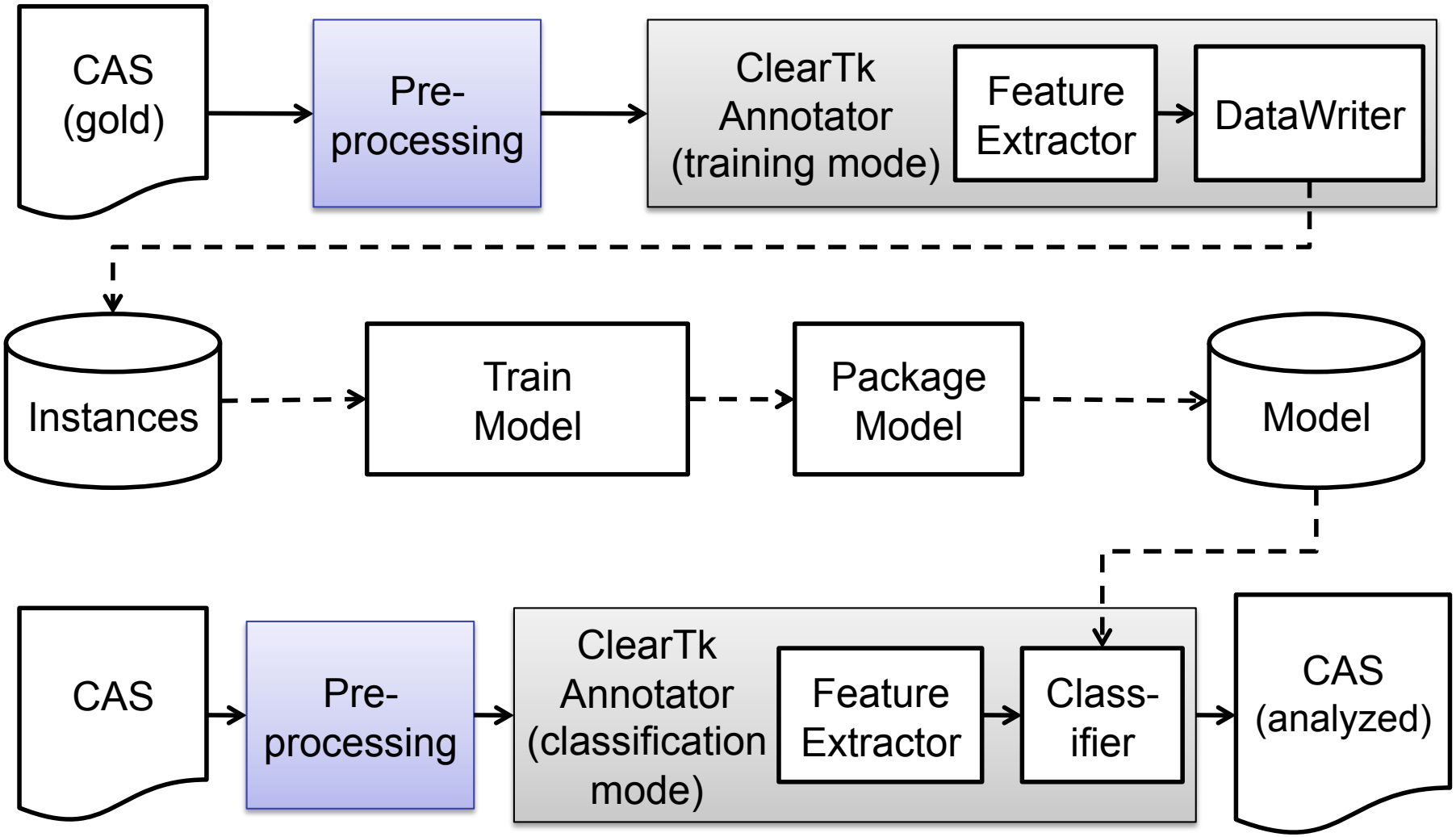
- What ClearTK-ML can do for me?
- What are its concepts?
- How do I use it?

ClearTK-ML Workflow

TECHNISCHE UNIVERSITÄT DARMSTADT

- Prepare training input
  - Extract of features from the CAS
  - Generate instances
  - Run pipeline
- Train
  - Train classifier
  - Package model as a JAR
- Use
  - Use model in an analysis task

Clear TK Workflow



Feature Extractors

TECHNISCHE UNIVERSITÄT DARMSTADT

- Prepare training input
  - **Extract of features from the CAS**
  - Generate instances
  - Run pipeline

Feature Extractor (anno.→feat.)

- TypePathExtractor
- SpannedTextExtractor
- OutcomeFeatureExtractor
- …

Feature Proliferator (feat.→feat.)

- LowerCaseProliferator
- NumericTypeProliferator
- CharacterNGramProliferator
- …

POS Tagging – Feature Extraction

TECHNISCHE UNIVERSITÄT DARMSTADT

```
// a list of feature extractors that require only the token: the stem of the word, the text
// of the word itself, plus features created from the word text like character ngrams
tokenFeatureExtractors = asList(
    new TypePathExtractor(Token.class, "stem/value"),
    new ProliferatingExtractor(
        new SpannedTextExtractor(),
        new LowerCaseProliferator(),
        new CapitalTypeProliferator(),
        new NumericTypeProliferator(),
        new CharacterNGramProliferator(RIGHT_TO_LEFT, 0, 2),
        new CharacterNGramProliferator(RIGHT_TO_LEFT, 0, 3)));

// a list of feature extractors that require the token and the sentence
contextFeatureExtractors = asList(new ContextExtractor<Token>(
    Token.class, new TypePathExtractor(Token.class, "stem“),
    new Preceding(2), new Following(2)));
```

Generating Instances

TECHNISCHE UNIVERSITÄT DARMSTADT

- Prepare training input
  - Extract of features from the CAS
  - **Generate instances**
  - Run pipeline

Instance<OUTCOME_TYPE>

OUTCOME_TYPE

- String
- Boolean
- Integer

SequenceDataWriterFactory

- DefaultMalletCrfDataWriterFactory
- DefaultGrmmDataWriterFactory
- ViterbiDataWriterFactory

DataWriterFactory

- DefaultMaxEntDataWriterFactory
- DefaultMalletDataWriterFactory
- …

POS Tagging – Generating Instances

TECHNISCHE UNIVERSITÄT DARMSTADT

```
for (Sentence sentence : select(jCas, Sentence.class)) {
    List<Instance<String>> instances = new ArrayList<Instance<String>>();
    List<Token> tokens = selectCovered(jCas, Token.class, sentence);

    for (Token token : tokens) {
        Instance<String> instance = new Instance<String>();

        for (SimpleFeatureExtractor extractor : this.tokenFeatureExtractors)
            instance.addAll(extractor.extract(jCas, token));

        for (ContextExtractor<Token> extractor : this.contextFeatureExtractors)
            instance.addAll(extractor.extractWithin(jCas, token, sentence));

        instance.setOutcome(token.getPos().getPosValue());

        // add the instance to the list
        instances.add(instance);
    }

    this.dataWriter.write(instances);
}
```

Only these lines will change for doing classification!

POS Tagging – Preprocessing Pipeline

TECHNISCHE UNIVERSITÄT DARMSTADT

- Prepare training input
  - Extract of features from the CAS
  - Generate instances
  - **Run pipeline**

```
runPipeline(
    createDescription(NegraExportReader.class, ...),
    createPrimitiveDescription(SnowballStemmer.class),
    createPrimitiveDescription(ExamplePosAnnotator.class,
        PARAM_DATA_WRITER_FACTORY_CLASS_NAME,
            DefaultMalletCRFDataWriterFactory.class.getName(),
        PARAM_OUTPUT_DIRECTORY,  MODEL_DIRECTORY));
```

POS Tagging – Preprocessing Pipeline

TECHNISCHE UNIVERSITÄT DARMSTADT

- Prepare training input
  - Extract of features from the CAS
  - Generate instances
  - **Run pipeline**

```
runPipeline(
    createDescription(NegraExportReader.class, ...),
    createPrimitiveDescription(SnowballStemmer.class),
    createPrimitiveDescription(ExamplePosAnnotator.class,
        PARAM_DATA_WRITER_FACTORY_CLASS_NAME,
            ViterbiDataWriterFactory.class.getName(),
        PARAM_OUTPUT_DIRECTORY,  MODEL_DIRECTORY,
        PARAM_DELEGATED_DATA_WRITER_FACTORY_CLASS,
            DefaultMaxentDataWriterFactory.class.getName(),
        PARAM_OUTCOME_FEATURE_EXTRACTOR_NAMES, new String[]
            { DefaultOutcomeFeatureExtractor.class.getName() })));
```

Training

TECHNISCHE UNIVERSITÄT DARMSTADT

- Train model based in the instances
- DataWriterFactory determines which ClassifierBuilder (algorithm) is used
  - Stored as metadata file in the model directory
- Hyperparameters can be passed to trainClassifier()
  - Depends on algorithm/library (look at the source Luke)
  - e.g. *iterations* and *cutoff* when using OpenNLP MaxEnt
- Package trained model as a convenient JAR

```
File dir = new File("MODEL_DIR");
JarClassifierBuilder<?> classifierBuilder =
    JarClassifierBuilder.fromTrainingDirectory(dir);
classifierBuilder.trainClassifier(dir, new String[0]);
classifierBuilder.packageClassifier(dir);
```

Performing Classification

TECHNISCHE UNIVERSITÄT DARMSTADT

```
for (Sentence sentence : select(jCas, Sentence.class)) {
    List<Instance<String>> instances = new ArrayList<Instance<String>>();
    List<Token> tokens = selectCovered(jCas, Token.class, sentence);

    for (Token token : tokens) {
        Instance<String> instance = new Instance<String>();

        for (SimpleFeatureExtractor extractor : this.tokenFeatureExtractors)
            instance.addAll(extractor.extract(jCas, token));

        for (ContextExtractor<Token> extractor : this.contextFeatureExtractors)
            instance.addAll(extractor.extractWithin(jCas, token, sentence));

        instance.setOutcome(token.getPos().getPosValue());

        // add the instance to the list
        instances.add(instance);
    }

    List<String> labels = classify(instances);
}
```

Only these lines changed from generating training data!

Machine Learning Tutorial

CB, GS, REC

Section 5b
ClearTK + The Lab = ML Experiments

Machine Learning Tutorial for the UKP lab
June 10, 2011

What to learn?

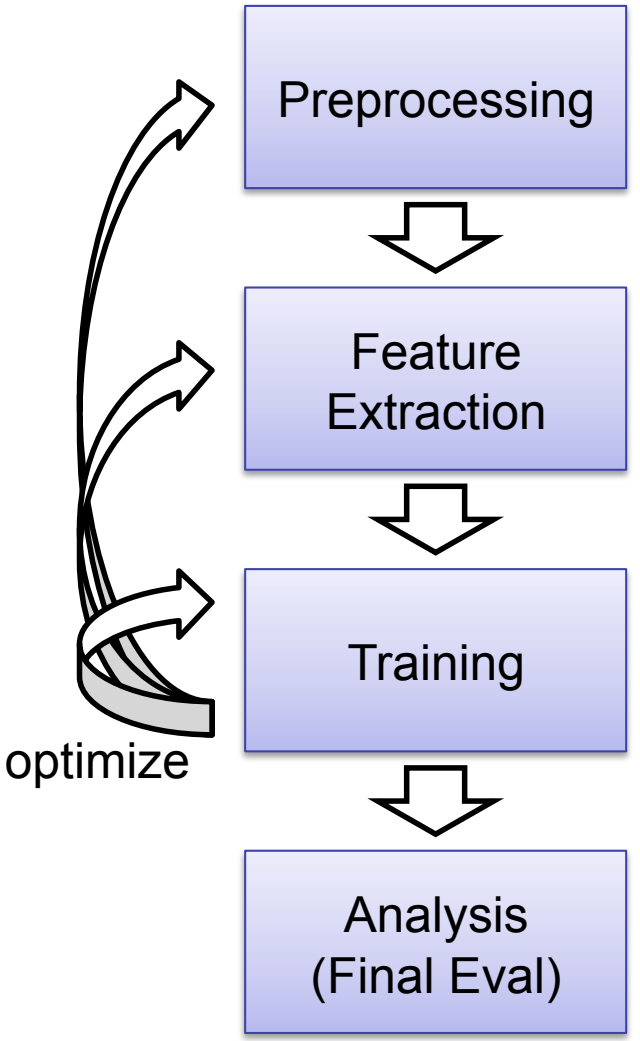
TECHNISCHE UNIVERSITÄT DARMSTADT

- Identify tasks
- Model tasks using the Lab
- Using the Lab to try different experiment parameters

Tasks

TECHNISCHE UNIVERSITÄT DARMSTADT

- Preprocessing task
  - Consumes corpus
  - Produces preprocessed XMI files
  - Split into training set and test set (not shown)
- Feature extraction task
  - Consumes preprocessed XMI files
  - Produces instances
- Training task
  - Consumes instances
  - Produces classifier
  - Evaluates classifier performance (not shown)
- Analysis task
  - Perform final evaluation (not shown)
- Repeat to optimize (not shown)



UKP

Experimental Setup

TECHNISCHE UNIVERSITÄT DARMSTADT

```
Task preprocessingTask = new UimaTaskBase() {...
Task featureExtractionTask = new UimaTaskBase() {...
Task trainingTask = new ExecutableTaskBase() {...
Task analysisTask= new UimaTaskBase () {...

ParameterSpace pSpace = new ParameterSpace(
    Dimension.create("corpusPath", CORPUS_PATH),
    Dimension.create("iterations", 20, 50, 100),
    Dimension.create("cutoff", 5));

featureExtractionTask.addImportLatest("XMI", "XMI", preprocessingTask.getType());
trainingTask.addImportLatest("MODEL", "MODEL", featureExtractionTask.getType());
analysisTask.addImportLatest("MODEL", "MODEL", trainingTask.getType());

BatchTask batch = new BatchTask();
batch.setParameterSpace(pSpace);
batch.setExecutionPolicy(ExecutionPolicy.USE_EXISTING);
batch.addTask(preprocessingTask);
batch.addTask(featureExtractionTask);
batch.addTask(trainingTask);
batch.addTask(analysisTask);
Lab.getInstance().run(batch);
```

Preprocessing Task

TECHNISCHE UNIVERSITÄT DARMSTADT

```
Task preprocessingTask = new UimaTaskBa
    @Discriminator String corpusPath;
    { setType("Preprocessing"); }

    CollectionReaderDescription getCollectionReaderDescription(TaskContext aCtx) {
        return createReader(NegraExportReader.class,
            NegraExportReader.PARAM_INPUT_FILE, corpusPath,
            NegraExportReader.PARAM_LANGUAGE, "de");
    }

    AnalysisEngineDescription getAnalysisEngineDescription(TaskContext aCtx) {
        File xmiDir = aCtx.getStorageLocation("XMI", READWRITE);
        return createEngine(
            createEngine(SnowballStemmer.class),
            createEngine(XmiWriter.class,
                XmiWriter.PARAM_PATH, xmiDir.getPath(),
                XmiWriter.PARAM_COMPRESS, true));
    }
};
```

```
ParameterSpace pSpace = new ParameterSpace(
    Dimension.create("corpusPath",
CORPUS_PATH),
    Dimension.create("iterations", 20, 50, 100),
    Dimension.create("cutoff", 5));
```

Feature Extraction Task

TECHNISCHE UNIVERSITÄT DARMSTADT

```
Task featureExtractionTask = new UimaTaskBase() {
  { setType("FeatureExtraction"); }

  CollectionReaderDescription getCollectionReaderDescription(TaskContext aCtx)  {
    File xmiDir = aCtx.getStorageLocation("XMI", READONLY);
    xmiDir.mkdirs();
    return createReader(XmiReader.class,
      XmiReader.PARAM_PATH, xmiDir.getPath(),
      XmiReader.PARAM_PATTERNS, new String[] { "[+]**/*.xmi.gz" });
  }

  AnalysisEngineDescription getAnalysisEngineDescription(TaskContext aCtx) {
    File md = aCtx.getStorageLocation("MODEL", READWRITE);
    md.mkdirs();
    return createEngine(
    createPrimitiveDescription(ExamplePosAnnotator.class,
      ExamplePosAnnotator.PARAM_DATA_WRITER_FACTORY_CLASS_NAME,
        ViterbiDataWriterFactory.class.getName(),
      ViterbiDataWriterFactory.PARAM_OUTPUT_DIRECTORY, md.getPath(),
      ViterbiDataWriterFactory.PARAM_DELEGATED_DATA_WRITER_FACTORY_CLASS,
        DefaultMaxentDataWriterFactory.class.getName())); }};
```

Training Task

TECHNISCHE UNIVERSITÄT DARMSTADT

```
Task trainingTask = new ExecutableTaskBase() {
    @Discriminator int iterations;
    @Discriminator int cutoff;

    { setType("TrainingTask"); }

    void execute(TaskContext aContext) {
        File dir = aContext.getStorageLocation("MODEL", READWRITE);
        JarClassifierBuilder<?> classifierBuilder =
            JarClassifierBuilder.fromTrainingDirectory(dir);
        classifierBuilder.trainClassifier(dir, new String[] {
            String.valueOf(iterations), String.valueOf(cutoff)});
        classifierBuilder.packageClassifier(dir);
    }
};
```

```
ParameterSpace pSpace = new ParameterSpace(
    Dimension.create("corpusPath", CORPUS_PATH),
    Dimension.create("iterations", 20, 50, 100),
    Dimension.create("cutoff", 5));
```

Analysis Task

TECHNISCHE UNIVERSITÄT DARMSTADT

```
Task analysisTask = new UimaTaskBase() {
  { setType("AnalysisTask"); }

  CollectionReaderDescription getCollectionReaderDescription(TaskContext aCtx) {
    return createDescription(TextReader.class,
      TextReader.PARAM_PATH, "src/test/resources/text",
      TextReader.PARAM_PATTERNS, new String[] { "[+]**/*.txt" },
      TextReader.PARAM_LANGUAGE, "de");
  }

  AnalysisEngineDescription getAnalysisEngineDescription(TaskContext aCtx) {
    File mod = new File(aCtx.getStorageLocation("MODEL", READONLY), "model.jar");
    File tsv = new File(aContext.getStorageLocation("TSV", READWRITE), "out.tsv");
    return createEngine(
      createPrimitiveDescription(StanfordSegmenter.class),
      createPrimitiveDescription(SnowballStemmer.class),
      createPrimitiveDescription(ExamplePosAnnotator.class,
        GenericJarClassifierFactory.PARAM_CLASSIFIER_JAR_PATH, mod.getPath()),
      createPrimitiveDescription(ImsCwbWriter.class,
        ImsCwbWriter.PARAM_OUTPUT_FILE, tsv.getAbsolutePath()));
  }};
```

Adding ML support to the Lab

TECHNISCHE UNIVERSITÄT DARMSTADT

- Provide serializable data models
  - E.g. for evaluation data
  - Simply any class that can read/write its state to a stream
- Provide Probes (e.g. CasConsumers) that evaluate against gold standard
  - Capture evaluation data
  - Serialize evaluation data do disk
- Provide Reports that nicely render evaluation results
  - Unserialize evaluation data from disk
  - Generate graphs, confusion matrix, …
- Lab has such things for IR already, but not for ML
  - Support to generate CSV files, Excel files, scalable PDF graphs is present

Thanks

- ClearTK Tutorial
  - http://code.google.com/p/cleartk/wiki/Tutorial
- ClearTK + The Lab Example
  - http://code.google.com/p/dkpro-lab/source/browse/#git%2Fde.tudarmstadt.ukp.dkpro.lab%2Fde.tudarmstadt.ukp.dkpro.lab.ml.example