Train Annotation Identifiers

TokensSequenceSolution
TrainingWindow (Training Window):
«Class»

Creates the window for a user interface whereby the user can determine the type of model to use for identifying sequences for annotation and the concepts that they map to, and allows user to set the options used during training and where to save the Solution object containing the datastructures generated during training

Responsibilities:
- Instantiate a type of TokenSequenceSolutionTrainer, the type depending on settings in the window
- Start trainers (via generate or update solution)
- Save solutiions generated by trainers

void saveRaptatSolution(RaptatSolution)

TokenSequenceSolutionTrainer:
«Interface»

Creates a TokenSequenceFinderSolution based on a set of documents and their annotations.

TokenSequenceSolution generateSolution(trainingData)

implements implements

ProbabilisticSolution Trainer:
«Class»

CRFSolution Trainer:
«Class»

TokensSequenceSolutionUpdater:
«Interface»

Takes an existing solution and updates it based on new training data

TokenSequenceIDSolution updateSolution(trainingData, RaptatSolution)

implements implments

ProbabilisticSolutionUpdater:
«Class»

CRFSolutionUpdater:
«Class»

Annotate Using Finders & Taggers

AnnotationWindow:
«Class»

Responsibilities:
- Build list of AnnotationGroup instances for annotation
- Load RaptatSolution
- Instantiate Annotator & start it
- Export List<AnnotatedPhrase> instances to Knowtator or eHOST format

void saveRaptatSolution(RaptatSolution)

Annotator:
«Class»

Responsibilities:
- Instantiate TokenSequenceFinder
- Instantiate AttributeTagger
- Identify annotated phrases adding and subtracting based on FacilitatorBlockers, AttributeFilters, and ContextFilters

void annotate(List of AnnotationGroups)

TokensSequenceFinder:
«Interface»

Identifies token sequences that map to concepts of interest using an TokenSequenceFinderSolution

List<AnnotatedPhrase> identifySequences(AnnotatedPhrase);
List<AnnotatedPhrase> identifySequences(List<AnnotatedPhrase>);

implements implements

CRFTSFinder:
«Class»

ProbabilisticTSFinder:
«Class»

CSAttributeTagger:
«Class»

Assigns attributes to annotated phrases based on a cue and scope model where the cue is a token or token sequence providing context to the words around it for a certain distance (the scope).

List<AnnotatedPhrase> assignAttribute(AnnotatedPhrase);

Solutions and Solution Subclasses

RaptatAnnotationSolution:
«Class»

Responsibilities:
- Hold data strucutures used by Annotator
- Data structures consist of:
  - + *TokenSequenceSolution (CRFSolution or ProbabilisticSolution)*
  - + *HashMap of AttributeTaqgerSolutions*
  - + *FacilitatorBlockerSolution*
  - + *AttributeFilters (HashSet?)*
  - + *ContextFilters (HashSet?) (e.g. context = 'in medlist')*

getters and setters for each field

TokenSequenceFinderSolution:
«Abstract»

Responsibilities:
- Data strucutures/Objects used by TokenSequenceFinders to find token sequences that should be mapped to particular concepts

getters and setters for each field

«extends» «extends»

CRFTSFinderSolution:
«Class»

ProbabilisticTSFinderSolution:
«Class»

AttributeTaggerSolution:
«Class»

Responsibilities:
- Hold data strucutures used by AttributeTagger
- Data structures consist of:
  - + *For CSAttributeTagger, the datastructures consist of CRF models to detect cue and scope*

getters and setters for each field

Train Attribute Taggers

AttributeTaggerTrainingWindow:
«Class»

Responsibilities:
- Instantiate crf trainer
- Instantiate crf updater
- Start trainers (via generate or update soluti
- Save solutiions generated by trainers

void saveRaptatSolution(RaptatSolution)

AttributionTrainer:
«Interface»

AttributionSolution generateSolution(trainingData)

implements

CSAttributionTrainer:
«Class»

CSAttributionSolution generateSolution(trainingData)

AttributionSolutionUpdater:
«Interface»

CSAnnotaotrSolution updateSolution(trainingData, CSAnnotatorSolu