



SEMANTIC SEARCH 3000

LSA-BASED RESEARCH TOOL FOR INFORMATION AND DOCUMENT RETRIEVAL

TERRANCE TAUBES (T14)

PROFESSOR BENAJIBA

CSCI 3907/6907 INTRO TO STATISTICAL NLP

GOAL:

To design a research tool that allows students to find the relevant documents within a large collection of text in order to facilitate a quick and direct approach to finding the appropriate information and sources for their research.

Designed to:

- Reduce the time needed for research
- Quickly find the documents most relevant to the information that the user seeks
- Allow students to organize their documents, discern which documents are of interest, and promptly access the text via the user interface

SEMANTIC SEARCH 3000: OVERVIEW

- The Semantic Search 3000 application is a tool that utilizes an array of natural language processing techniques to compute a similarity score between a user's search query and each document within a specified document group.
- Users are able to organize collections of text files into Document Groups, which are directories containing the text files that are to be grouped together.
- Users are then able to enter search queries into the application and find the most relevant documents within the current Document Group.
- Users interact with the Semantic Search 3000 application using its graphical user interface.

API & MODULES

LSA Model Generation

- Gensim

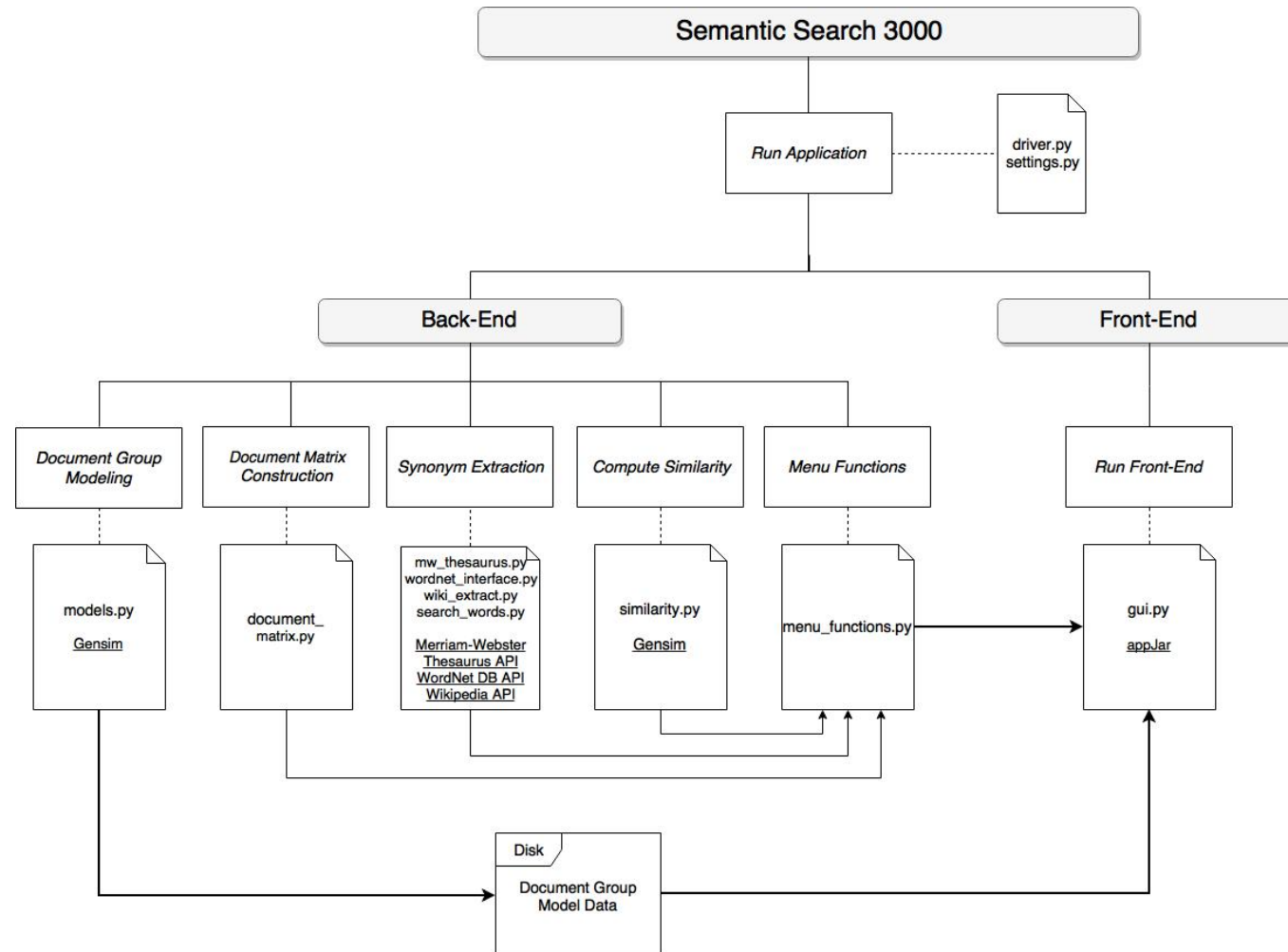
Synonym Extraction

- Merriam-Webster Thesaurus API
- WordNet Database API
- Wikipedia API

Graphical User Interface

- appJar

SEMANTIC SEARCH 3000: DESIGN



APPLICATIONS OF NLP

- Regular Expressions
- Text Normalization (Data Wrangling, Tokenization, Segmentation, Lemmatization)
- External Lexicon and Thesaurus APIs
- Information Extraction and Retrieval
- Latent Semantic Analysis
- Term-Document and Word-Word Matrices
- Term Frequency-Inverse Document Frequency

SEMANTIC SEARCH 3000: FUNCTIONS

Semantic Search 3000 has 3 Major Functions:

1. Search
2. Select Documents
3. Upload Documents

UPLOAD DOCUMENTS

- The 'Upload Documents' function allows users to specify a directory of text files as input, which is then used to build a LSA model for the directory and form the Document Group.

SELECT DOCUMENTS

- The 'Select Documents' function allows users to specify which Document Group they would like to use, and then loads the Document Group's model data.

ISSUES WITH INFORMATION RETRIEVAL

- Synonymy – Multiple ways to express or describe the meaning of the same thing
 - Search query words may not be found within a document even though the document is relevant
 - Need a way to include relevant search terms
- Compound Search Terms – “New York”, “Shake Shack”, “Machine Learning”
 - Search results find matches based on individual word matches and not matches of the whole concept
 - Query = “candy apple”, Doc1 = {“candy” : 9, “apple” : 0, “candy apple” : 0}, Doc2 = {“candy” : 1, “apple” : 1, “candy apple” : 1}
 - Need a way to add to the similarity scores for documents that contain compound matches

SEARCH

Search Pipeline:

1. Get user query
2. Preprocess query (lowercase, RegEx to remove punctuation and clitics, lemmatization)
3. Get list of synonyms for query words from APIs.
4. Build Term-Document Matrices for query words and API synonyms.
5. Get Word-Word Matrix counts
6. [Similarity Scoring Function]
7. Sort Documents by Relevancy

LATENT SEMANTIC ANALYSIS

- Latent Semantic Analysis (LSA) is a language processing technique that is able to find the semantic, or underlying meanings of text and to represent these semantic values in the form of vectors.
- Similar words and topics will be represented by the LSA model with similar vectors.
- Words and topics appearing within similar contexts will also be represented with similar vectors.
 - *Synonymy*

LATENT SEMANTIC ANALYSIS

- The similarity between two vector representations can be computed by taking the cosine of the vectors, returning a value between $(-1, +1)$, with a value of $+1$ meaning the vectors are identical.
- The **foundation** of the similarity scores computed between queries and documents is based on the cosine similarity value taken between the vector representation of the search query and the vector representation of the Document Group.
- Using LSA as the foundation of the similarity scoring helps give high weightings towards documents found to be semantically similar and low weightings to documents that are not, essentially eliminating the irrelevant documents from the search at the beginning.

QUERY WORD-DOCUMENT TITLE MATCHING

- The Query Word-Document Title matching function adds positive weighting to documents that have query words within their titles.

TERM-DOCUMENT MATRIX & TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY

- A Term-Document Matrix is composed of search words as row entries and documents as column entries.
- The cells for each corresponding (word, document) pair contain the frequency for that word in that particular document.
- The frequencies of query word matches is summed for each document (the values in each document column are summed up) and then divided by that document's total number of tokens, returning the TF-IDF values that are to be added to each document's similarity score.

WORD-WORD MATRIX

- A Word-Word matrix is a matrix with both the rows and columns represented by the query words.
- A Word-Word matrix is constructed for each document in the Document Group.
- The cells for each corresponding (word, word) pair contain the number of times each pair of query words appears within a document.
 - *Compound Search Terms*
- All the values within a document's Word-Word matrix are summed and divided by the total number of query word combinations, returning the Word-Word frequency values that are to be added to the document's similarity score.

SIMILARITY SCORING FUNCTION (HIGH-LEVEL DESCRIPTION)

for (doc in Document Group):

doc_score = LSA_Similarity(query words, Document Group model) / float(2)

doc_score += Title_Similarity(query words) / float(2)

doc_score += TFIDF_Similarity(query word Term-Document matrix) / float(2)

doc_score += TFIDF_API_Similarity(API synonyms Term-Document matrix) / float(6)

doc_score += WW_Similarity(query word Word-Word matrix) / float(5)

BACK-END OUTPUT

Results between Vector Space of the Query and Vector Space of the Document Group

```
[195, 'Language technology.txt', 0.97754514]
[121, 'Noam Chomsky.txt', 0.97013515]
[234, 'Alan Turing.txt', 0.96761435]
[61, 'IBM.txt', 0.96699286]
[63, 'Natural language understanding.txt', 0.96594411]
[237, 'Corpus linguistics.txt', 0.96334791]
[69, 'Natural language.txt', 0.96215558]
[107, 'Natural speech.txt', 0.96215558]
[148, 'History of natural language processing.txt', 0.96088153]
[172, 'Journal of Machine Learning Research.txt', 0.96061355]
[47, 'Inductive programming.txt', 0.95817387]
[99, 'Computer-assisted translation.txt', 0.95809698]
[97, 'Machine ethics.txt', 0.95332015]
[52, 'Spoken dialogue system.txt', 0.95318896]
[142, 'World Wide Web.txt', 0.9527775]
[178, 'List of important publications in computer science.txt', 0.95178699]
[100, 'Speech synthesis.txt', 0.95168352]
[109, 'Text-to-speech.txt', 0.95168352]
[208, 'List of natural language processing toolkits.txt', 0.95124704]
[238, 'Internet.txt', 0.95053452]
```

Results of matching Query words within Document titles:

```
['Statistical machine translation.txt', 1.0]
['Rule-based machine translation.txt', 0.6666666666666666]
['Example-based machine translation.txt', 0.6666666666666666]
['Machine translation.txt', 0.6666666666666666]
['Brain machine interface.txt', 0.3333333333333333]
['Automatic translation.txt', 0.3333333333333333]
['Journal of Machine Learning Research.txt', 0.3333333333333333]
['Azure machine learning studio.txt', 0.3333333333333333]
['Machine Learning (journal).txt', 0.3333333333333333]
['Machine learning.txt', 0.3333333333333333]
['Statistical models.txt', 0.3333333333333333]
['Statistical inference.txt', 0.3333333333333333]
['Computer-assisted translation.txt', 0.3333333333333333]
['Machine ethics.txt', 0.3333333333333333]
['Amazon Machine Learning.txt', 0.3333333333333333]
['Machine learning control.txt', 0.3333333333333333]
['List of machine learning algorithms.txt', 0.3333333333333333]
['List of datasets for machine learning research.txt', 0.3333333333333333]
['International Conference on Machine Learning.txt', 0.3333333333333333]
['Machine perception.txt', 0.3333333333333333]
```

BACK-END OUTPUT

```
Results of the term frequency-inverse document frequency calculation for Query words
:
[222, 'Example-based machine translation.txt', 0.074509803921568626]
[201, 'Statistical machine translation.txt', 0.061801446416831031]
[17, 'Machine perception.txt', 0.055384615384615386]
[99, 'Computer-assisted translation.txt', 0.054223149113660066]
[191, 'Automatic translation.txt', 0.053216103655714948]
[19, 'Machine translation.txt', 0.053216103655714948]
[16, 'Computational statistics.txt', 0.047619047619047616]
[172, 'Journal of Machine Learning Research.txt', 0.035714285714285712]
[9, 'Stochastic grammar.txt', 0.031531531531531529]
[24, 'International Conference on Machine Learning.txt', 0.029268292682926831]
[148, 'History of natural language processing.txt', 0.029126213592233011]
[125, 'Statistical models.txt', 0.026554856743535988]
[230, 'Rule-based machine translation.txt', 0.025000000000000001]
[147, 'Mlpy.txt', 0.022950819672131147]
[120, 'Statistical inference.txt', 0.022288261515601784]
[151, 'Machine Learning (journal).txt', 0.022034996759559299]
[134, 'Machine learning.txt', 0.022034996759559299]
[58, 'Amazon Machine Learning.txt', 0.022034996759559299]
[39, 'Active learning (machine learning).txt', 0.022034996759559299]
[32, 'List of machine learning algorithms.txt', 0.022034996759559299]
Results of the term frequency-inverse document frequency calculation for API words
[195, 'Language technology.txt', 0.012903225806451613]
[172, 'Journal of Machine Learning Research.txt', 0.01020408163265306]
[16, 'Computational statistics.txt', 0.0090702947845804991]
[3, 'Computation.txt', 0.0088495575221238937]
[122, 'Handwriting recognition.txt', 0.0078175895765472316]
[231, 'Speech corpus.txt', 0.0071684587813620072]
[177, 'Predictive text.txt', 0.0068352699931647299]
[107, 'Natural speech.txt', 0.0063965884861407248]
[69, 'Natural language.txt', 0.0063965884861407248]
[63, 'Natural language understanding.txt', 0.0056497175141242938]
[98, 'Word2vec.txt', 0.0050403225806451612]
[92, 'Parsing.txt', 0.0050038491147036184]
[223, 'Relationship extraction.txt', 0.0050000000000000001]
[14, 'Automated theorem proving.txt', 0.0048449612403100775]
[139, 'Computational learning theory.txt', 0.0047694753577106515]
[173, 'Evolutionary algorithm.txt', 0.0047483380816714148]
[82, 'Multi-label classification.txt', 0.0046728971962616819]
[159, 'Inductive bias.txt', 0.0046403712296983757]
[9, 'Stochastic grammar.txt', 0.0045045045045045045]
[40, 'Algorithm design.txt', 0.0044843049327354259]
```

BACK-END OUTPUT

Results of the co-occurrence counts per document, normalized by the number of Query word combinations

```
[237, 'Corpus linguistics.txt', 1.0]
[230, 'Rule-based machine translation.txt', 1.0]
[227, 'Computer science.txt', 1.0]
[222, 'Example-based machine translation.txt', 1.0]
[216, 'Automatic summarization.txt', 1.0]
[208, 'List of natural language processing toolkits.txt', 1.0]
[204, 'Hidden Markov models.txt', 1.0]
[201, 'Statistical machine translation.txt', 1.0]
[191, 'Automatic translation.txt', 1.0]
[188, 'Information extraction.txt', 1.0]
[178, 'List of important publications in computer science.txt', 1.0]
[171, 'Named entity recognition.txt', 1.0]
[156, 'Artificial intelligence.txt', 1.0]
[148, 'History of natural language processing.txt', 1.0]
[119, 'Latent semantic indexing.txt', 1.0]
[117, 'Deep learning.txt', 1.0]
[93, 'Word-sense disambiguation.txt', 1.0]
[91, 'Agglutination.txt', 1.0]
[88, 'Text corpus.txt', 1.0]
[78, 'Computational linguistics.txt', 1.0]
```

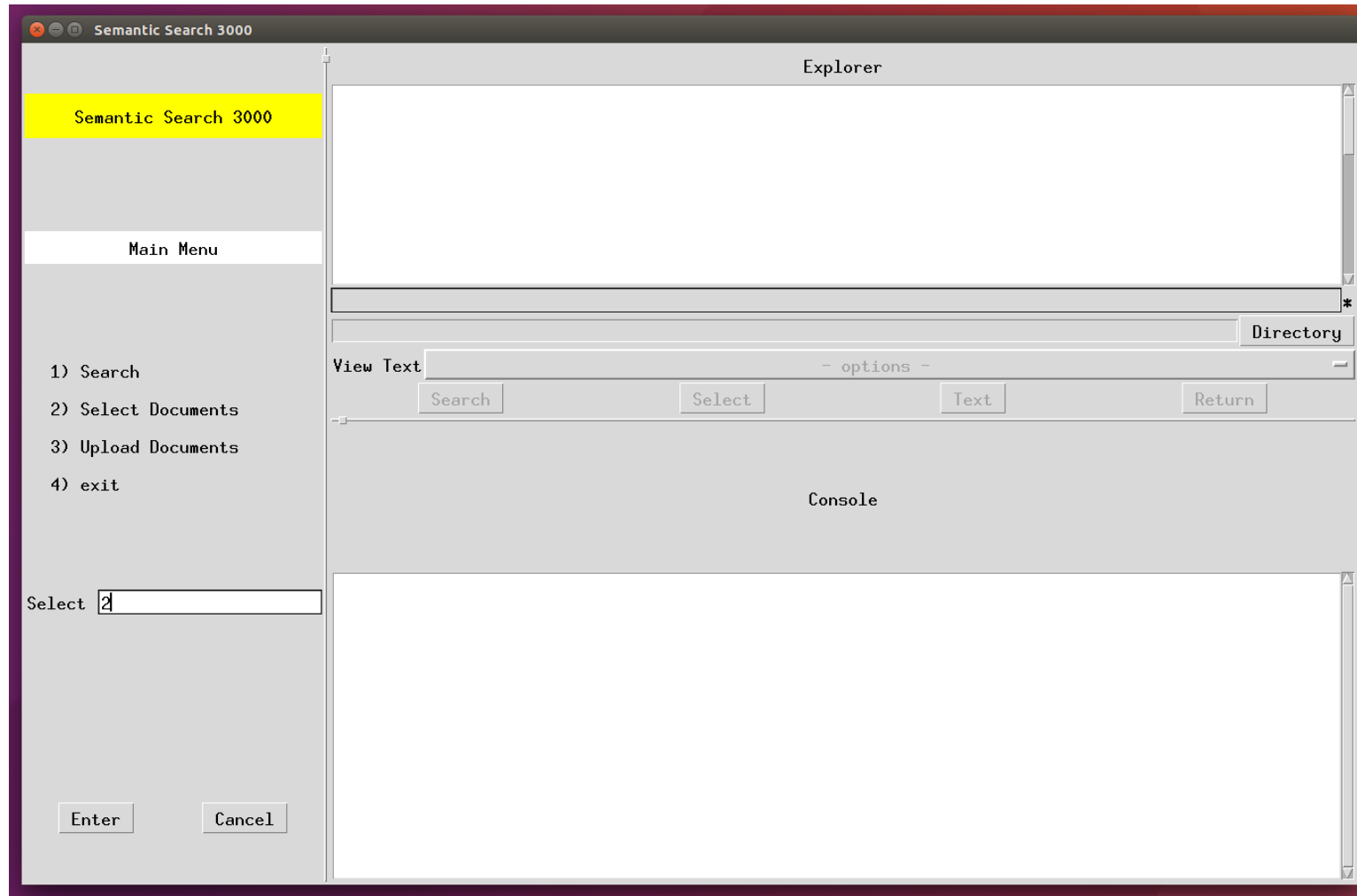
Documents by Calculated Score

```
[201, 'Statistical machine translation.txt', 1.0]
[222, 'Example-based machine translation.txt', 0.98538132480811447]
[19, 'Machine translation.txt', 0.95229741005893742]
[191, 'Automatic translation.txt', 0.9006409214296428]
[230, 'Rule-based machine translation.txt', 0.89966268347519518]
[148, 'History of natural language processing.txt', 0.81463835501932647]
[88, 'Text corpus.txt', 0.79036618945210468]
[208, 'List of natural language processing toolkits.txt', 0.78630731999014447]
[11, 'Terminology extraction.txt', 0.78298298124746868]
[156, 'Artificial intelligence.txt', 0.78140552903380522]
```

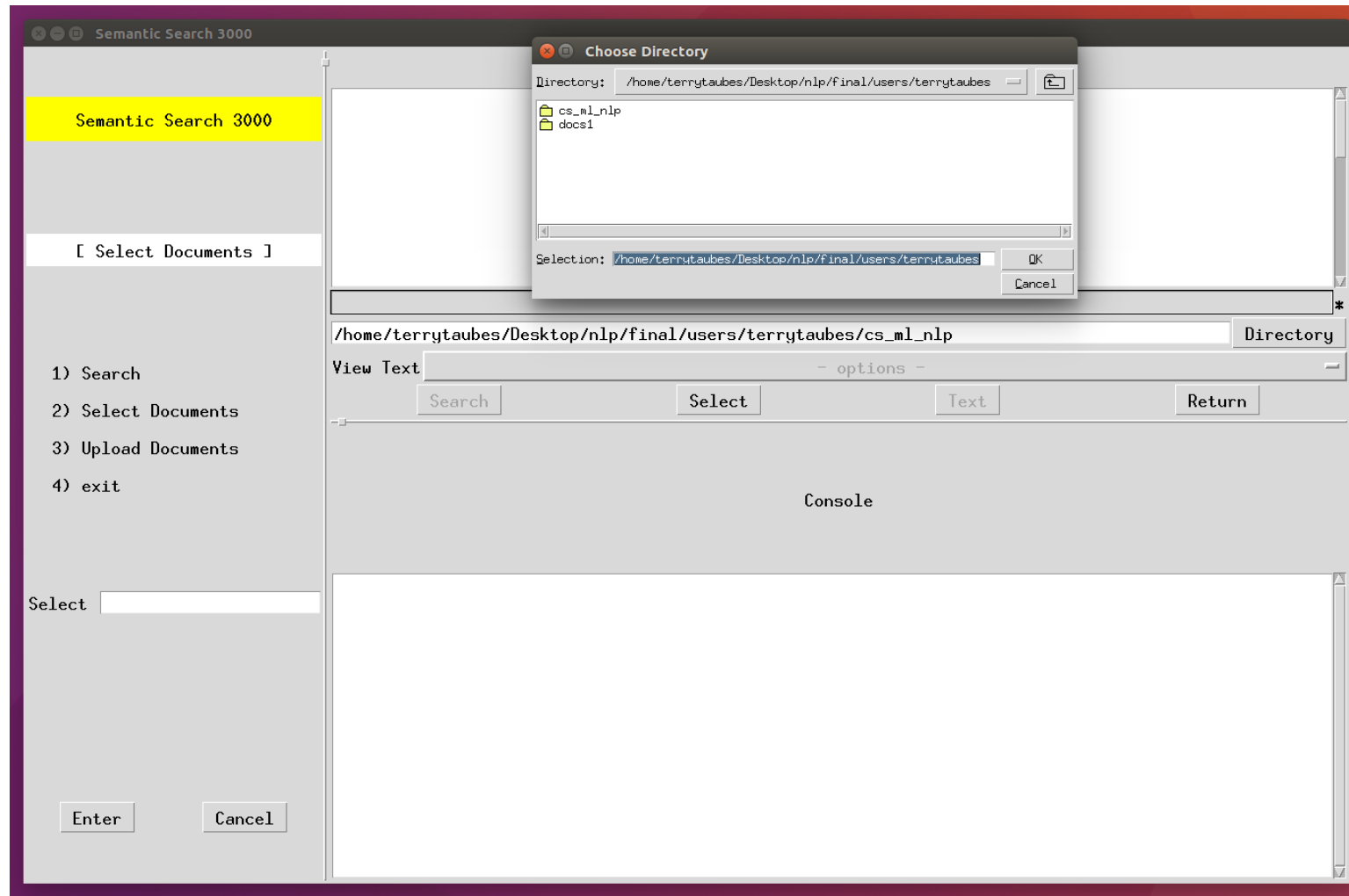
DEMO: LOGIN

The screenshot shows a window titled "Semantic Search 3000" with a dark red border. The interface is split into two main vertical sections. The left section is a sidebar with a yellow header containing the text "Semantic Search 3000". Below the header is a white button labeled "Login". At the bottom of the sidebar are two input fields: "Username" with the text "terrytaubes" and "Password" with a masked field of asterisks. Below these fields are two buttons: "Login" and "Cancel". The right section is a large white area with a scroll bar. It contains a search bar with the placeholder text "-- enter a directory --" and a "Directory" button. Below the search bar is a "View Text" section with a dropdown menu showing "- options -" and four buttons: "Search", "Select", "Text", and "Return".

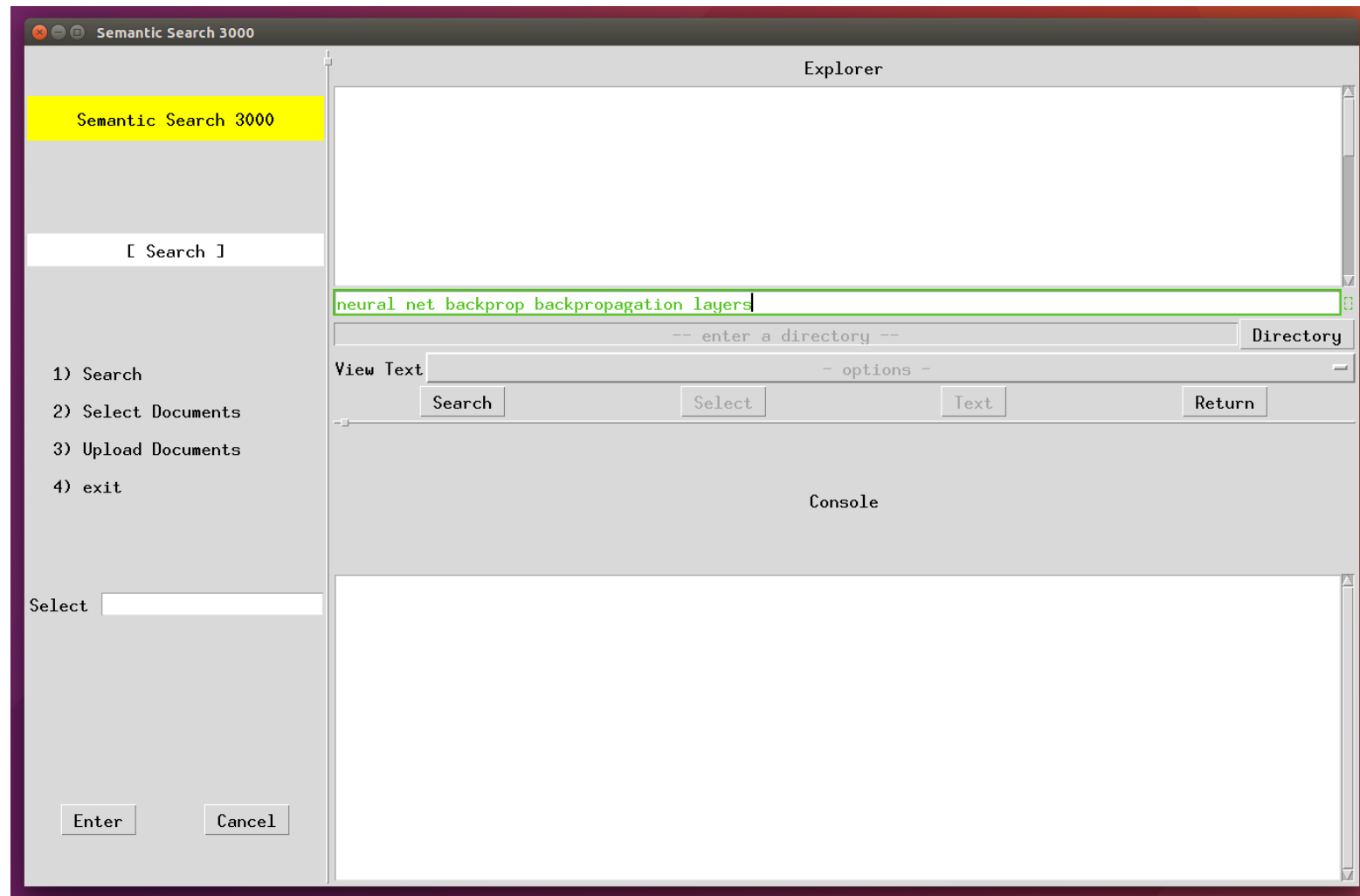
DEMO: MAIN MENU



DEMO: SELECT/UPLOAD DOCUMENTS



DEMO: SEARCH



DEMO: SEARCH RESULTS

The screenshot displays the 'Semantic Search 3000' application window. On the left, a sidebar contains a search bar with the text 'Semantic Search 3000', a search button labeled '[Search]', a list of menu items (1) Search, 2) Select Documents, 3) Upload Documents, 4) exit, a 'Select' input field, and 'Enter' and 'Cancel' buttons at the bottom.

The main area is divided into two sections. The top section, titled 'Explorer', shows a list of search results with their corresponding scores:

Rank	Document Name	Score
1.	Artificial neural network.txt	100.0
2.	Backpropagation.txt	98.5
3.	Convolutional neural network.txt	98.1
4.	ADALINE.txt	93.9
5.	Deep learning.txt	91.0
6.	Artificial intelligence.txt	86.3
7.	Multilayer perceptron.txt	61.8
8.	Artificial neuron.txt	59.9
9.	Speech recognition.txt	55.2
10.	List of important publications in computer science.txt	53.3

The bottom section, titled 'View Text', shows the selected document 'neural net backprop backpropagation layers' with a file path '-cs_ml_nlp-'. It includes buttons for 'Search', 'Select', 'Text', and 'Return'. Below this is a 'Console' area, which is currently empty.

DEMO: VIEW TEXT

The screenshot displays the Semantic Search 3000 application window. On the left, a sidebar contains a search bar with the text "[Search]", a list of actions (1) Search, 2) Select Documents, 3) Upload Documents, 4) exit, and a "Select" field with an input box. Below the sidebar are "Enter" and "Cancel" buttons. The main area is divided into three sections: "Explorer" at the top, "View Text" in the middle, and "Console" at the bottom. The Explorer shows a list of 10 files with their scores, with the first file selected. The View Text section shows the content of "Artificial neural network.txt" with "Search", "Select", "Text", and "Return" buttons. The Console section displays the text content of the selected file.

Rank	File Name	Score
1.	Artificial neural network.txt	100.0
2.	Backpropagation.txt	98.5
3.	Convolutional neural network.txt	98.1
4.	ADALINE.txt	93.9
5.	Deep learning.txt	91.0
6.	Artificial intelligence.txt	86.3
7.	Multilayer perceptron.txt	61.8
8.	Artificial neuron.txt	59.9
9.	Speech recognition.txt	55.2
10.	List of important publications in computer science.txt	53.3

neural net backprop backpropagation layers

Artificial neural networks (ANNs) or connectionist systems are computing systems inspired by the biological neural networks that constitute animal brains. Such systems learn (progressively improve performance) to do tasks by considering examples, generally without task-specific programming. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the analytic results to identify cats in other images. They have found most use in applications difficult to express in a traditional computer algorithm using rule-based programming.

An ANN is based on a collection of connected units called artificial neurons (analogous to biological neurons in an animal brain). Each connection (synapse) between neurons can transmit a signal to another neuron. The receiving (postsynaptic) neuron can process the signal(s) and then signal downstream neurons connected to it. In common ANN implementations, the synapse signal is simply a real number, and the output of each neuron is calculated by a non-linear function of the sum of all its input. Neurons and synapses may also have a weight that varies as learning proceeds, which can increase or decrease the strength of the signal that it sends downstream. Further, they may have a threshold such that only if the aggregate signal is below (or above) that level is the downstream signal sent.

SEMANTIC SEARCH 3000

Thank You!

SLIDES

- [2 - 3] S
- [4 - 5] T
- [6 - 9] S
- [10 - 18] T