Text Analyzer

Sarah El-Sherbiny, Takaki Hagen

Jun 06, 2019

CONTENTS: MODULES

CHAPTER

ONE

SRC

1.1 analyzeText module

analyzeText.**analyze_single_sentence**(*sentence*)
Analyze a single sentence and return a one dimensional matrix with scores for all attributes of this sentence.

Parameters **sentence** – the single sentence to analyze

Returns a score matrix *1 x m*, where *m* is the number of attributes

analyzeText.**calculate_nominal_form_score**(*sentence*)
Calculate the Nominal Forms (NF). This is the combination of the noun-to-verb ratio and the number of nominal forms. Nominal forms include gerunds, nominalized words and nouns. Nominalized words contain words with the endings *ing*, *ity*, *ness*, and similar.

Parameters **sentence** – the single sentence to analyze

Returns a value that represents the NF score of the sentence

analyzeText.**calculate_sentence_length_score**(*sentence*)
Calculate the Sentence Length (SL). This is defined by the number of words in a sentence.

Parameters **sentence** – the single sentence to analyze

Returns a value that represents the SL score of the sentence

analyzeText.**calculate_sentence_structure_score**(*sentence*)
Calculate the Sentence Structure (SS). The complexity is measured by branching in the sentence tree. It is increased when the sentence is interrupted by sub-sentences or parenthesis.

Parameters **sentence** – the single sentence to analyze

Returns a value that represents the SS score of the sentence

analyzeText.**calculate_vocabulary_complexity_score**(*sentence*)
Calculate the Vocabulary Complexity (VC). This is the percentage of terms not contained in a list of the *1000* most frequent terms in english language.

Parameters **sentence** – the single sentence to analyze

Returns a value that represents the VC score of the sentence

analyzeText.**calculate_word_length_score**(*sentence*)
Calculate the Word Length (WL). This is the average number of characters in a word.

Parameters **sentence** – the single sentence to analyze

Returns a value that represents the WL score of the sentence

analyzeText.**map_to_score**(*value*, *min_limit*, *max_limit*)
Map a value from *[min_limit max_limit]* to the interval *[0 1]* and return a score value.

Parameters

- **value** – the value to map
- **min_limit** – the min limit of the value
- **max_limit** – the max limit of the value

Returns the mapped score value in the interval *[0 1]*

analyzeText.**replace_punctuation**(*sentence*)
Replace the punctuation in a sentence

Parameters **sentence** – the sentence to process

Returns the processed sentence without spaces and without punctuation

analyzeText.**text_analysis**(*sentences*)
Analyze multiple sentences, set an annotation text for feature scores that hit a predefined limit and return a matrix with scores for all attributes.

Parameters **sentences** – all sentences to analyze

Returns a score matrix *n x m*, where *n* is the number of sentences and *m* is the number of attributes

1.2 mainWindow module

class mainWindow.**ItemDelegate**
Bases: PyQt5.QtWidgets.QStyledItemDelegate

Re-implement the edit event to detect changes in the detail view.

cellEditingStarted

createEditor(*parent*, *option*, *index*)
Re-implement function to customize the edit behavior.

class mainWindow.**Ui_MainWindow**
Bases: object

The main window of the user interface. It defines all ui elements.

cell_edit_start(*row*, *column*)
Called when the user selects a cell to save the cell data.

Parameters

- **row** – the row of the cell selected
- **column** – the column of the cell selected

Returns None

cell_edited()
Called when a cell in the detail view is changed. It updates all scores and colors of the sentence edited.

Returns None.

connect_buttons()
Connect the pdf buttons with their processing methods.

Returns None

static download_nltk()
Open the NLTK downloader.

Returns None

generate_pdf_with_button()
Generate a pdf, highlight it, and open it with the standard program of the user.

Returns None

static paint_text(*pixmap*, *start_pos*, *end_pos*, *start_text*, *end_text*, *color*)
Paint two text-strings on a pixmap.

Parameters

- **pixmap** – the pixmap to paint the text on
- **start_pos** – the position of the first text-string
- **end_pos** – the position of the second text-string
- **start_text** – the first text-string to paint
- **end_text** – the second text-string to paint
- **color** – the color used for painting

Returns the pixmap with the text painted on

process_file(*file_name*)
Initiate the sentence extraction and set the scores for all sentence data.

Parameters **file_name** – the name of the file selected

Returns None

resizeEvent(*event*)

resize_detail_view()
Resize the columns of the detail view.

Returns None

retranslateUi(*MainWindow*)
Retranslate the view of the ui.

Parameters **MainWindow** – the main window of the ui

Returns None

set_pdf()
Called when the user clicks the select-pdf button. It opens a pdf-file selection dialog. Further, it initiates the file processing.

Returns None

setupUi(*MainWindow*)
Setup all ui elements.

Parameters **MainWindow** – the main window of the ui

Returns None

testPrint()
Prints a message when the generate-pdf-button is clicked.

Returns None

mainWindow.**calculate_sentence_score**(*score_matrix*, *row*)
Calculate the score of a single sentence.

Parameters

- **score_matrix** – the matrix with the score values
- **row** – the row of the sentence, whose score is calculated

Returns the calculated score of the sentence

mainWindow.**interpolate_color**(*color2*, *color1*, *t*)
Interpolate between two colors.

Parameters

- **color2** – one color for the interpolation
- **color1** – another color for the interpolation
- **t** – the interpolation coefficient

Returns the interpolated color value

mainWindow.**interpolate_score_color**(*score*, *sentence*)
Interpolate between colors based on the feature score.

Parameters

- **score** – the score of the feature
- **sentence** – the sentence whose feature color is interpolated

Returns None

mainWindow.**interpolate_sentence_color**(*sentence_score*, *sentence*)
Interpolate between colors based on the sentence score.

Parameters

- **sentence_score** – the score of the sentence
- **sentence** – the sentence whose color is interpolated

Returns None

mainWindow.**nominal_form_slider_value_change**(*value*)
Called when the user changes the nominal form slider in the ui. This leads to an update of the sentence score values and of the sentence color values.

Parameters **value** – the updated value of the nominal form slider

Returns None

mainWindow.**sentence_length_slider_value_change**(*value*)
Called when the user changes the sentence length slider in the ui. This leads to an update of the sentence score values and of the sentence color values.

Parameters **value** – the updated value of the sentence length slider

Returns None

mainWindow.**sentence_structure_slider_value_change**(*value*)
Called when the user changes the sentence structure slider in the ui. This leads to an update of the sentence score values and of the sentence color values.

Parameters **value** – the updated value of the sentence structure slider

Returns None

mainWindow.**slider_count**()
Count how many sliders have an effect on the result. These are all sliders with a weight higher than zero.

Returns the number of sliders with a weight higher than zero

mainWindow.**update_colors**()
Update the colors of all sentences. First calculate an updated score value. Then interpolate the new sentence colors based on the scores calculated. Finally update the sentence colors.

Returns None

mainWindow.**update_overall_score**(*sum_score*, *num*)
Calculate an overall score for the document and show it in the ui.

Parameters

- **sum_score** – the sum of all sentence score values
- **num** – the number of sentences

Returns None

mainWindow.**update_score_color**(*row*, *column*, *sentence*)
Update the color of a feature.

Parameters

- **row** – the row of the feature to update
- **column** – the column of the feature to update
- **sentence** – the sentence whose feature color is updated

Returns None

mainWindow.**update_sentence_color**(*row*, *sentence*)
Update the color of a sentence.

Parameters

- **row** – the row of the sentence to update
- **sentence** – the sentence whose color is updated

Returns None

mainWindow.**vocabulary_slider_value_change**(*value*)
Called when the user changes the vocabulary slider in the ui. This leads to an update of the sentence score values and of the sentence color values.

Parameters **value** – the updated value of the vocabulary slider

Returns None

mainWindow.**word_length_slider_value_change**(*value*)
Called when the user changes the word length slider in the ui. This leads to an update of the sentence score values and of the sentence color values.

Parameters **value** – the updated value of the word length slider

Returns None

1.3 overlay module

class overlay.**Overlay**(*parent=None*)
Bases: PyQt5.QtWidgets.QWidget

Show a waiting indicator as an overlay.

Code taken from: https://wiki.python.org/moin/PyQt/A%20full%20widget%20waiting%20indicator

paintEvent(*event*)
Paint circles on screen as an waiting indicator for the user.

Parameters **event** – loading pdf

Returns None

showEvent(*event*)
Show the painted waiting indicator.

Parameters **event** – loading pdf

Returns None

timerEvent(*event*)
Timer for the duration of the inidicator

Parameters **event** – loading pdf

Returns None

1.4 pdfHandler module

class pdfHandler.**CharData**(*character*, *bb0*, *bb1*, *bb2*, *bb3*)
Bases: object

Hold properties of characters, which are extracted from the pdf data.

class pdfHandler.**PageData**
Bases: object

Hold the data of each page including sentences and characters.

addChar(*c*)
Add character data to the list.

Parameters **c** – Added character.

Returns None

addSentence(*s*)
Add sentence data to the list.

Parameters **s** – Added sentence.

Returns None

class pdfHandler.**PdfHandler**(*pdfPath*)
Bases: object

Handle whole pdf data.

generateHighlightedPdf()
Generate highlighted pdf with respect to each color and annotating text of it.

Returns None

getSentence()
Gets all sentences of the pdf data.

Returns Whole sentences.

makeSentence()
Make sentences from extracted characters.

Returns None

textExtracWithCoord()
Extract each character from pdf data. The character and its coordinates are extracted.

Returns None

class pdfHandler.**SentenceData**(*sentence*, *rectList*, *pageNum*)
Bases: object

setAnnotation(*annotText*)
Set annotate text for pdf. The text will be annotated in the pdf data.

Parameters **annotText** – Annotate text.

Returns None

setColor(*color*)
Set the annotation color for the sentence.

Parameters **color** – The color

Returns None

setRectList(*offset*)
Define original coordinate for pdf.

Parameters **offset** – The offsets from original coordinate.

Returns None

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX