Number of Syntactic Constituents: Adjectival Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of adjectival modifiers in the text. Gives an absolute number.

An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP Used also for numbers, when they represent age. John Smith (33) ...

Ex: "nuovo record" amod(record, nuovo) nell'ultima edizione della famosa maratona" amod(maratona, famosa)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Noun Compound Modifier

Counts the number of noun compound modifiers in the text. Gives an absolute number.

A noun compound modifier of an NP is any noun that serves to modify the head noun. (Note that in the current system for dependency extraction, all nouns modify the rightmost noun of the NP { there is no intelligent noun compound analysis. This is likely to be fixed once the Penn Treebank represents the branching structure of NPs.)

Ex: "la macchina cinema" nn(macchina, cinema); "l'effetto serra" nn(effetto, serra)

Tint ha dato: L'effetto serra è irreversibile.

Dependency Parse (enhanced plus plus dependencies): root(ROOT-0, irreversibile-5) det(effetto-2, L'-1) nsubj(irreversibile-5, effetto-2) compound(effetto-2, serra-3) cop(irreversibile-5, è-4) punct(irreversibile-5, .-6)

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

```
ParseTreeAnnotator.xml
```

POS Density Feature: Indicative Present Tense Verb

Calculates indicative imperfect tense verbs density of the text. Indicative imperfect tense verbs include for Italian: indicative imperfect : Vip Vip3 VAip VAip3 VMip VMip3.

Availability:

```
This feature is NOT available for: English, German
```

Formula:

```
VBDDensity = numVBDs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Complex T-units

Calculates the number of a specific syntactic constituents in the text. Counts the number of complex T-units in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Modifier

Calculates density of modifier the text. Modifier are adverbs and adjectives and include for English: the Penn Treebank tags JJ, JJR, JJS, RB, RBR, RBS, WRB for German: the Tiger tag ADV, ADJA, ADJD, for Italian: B BN As Ap An APs APp APn.

Formula:

```
modDensity = numModifier / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Direct Object

Calculates the number of a specific syntactic constituents in the text. Counts the number of direct objects in the text. Gives an absolute number.

The direct object of a VP is the noun phrase which is the (accusative) object of the verb

Ex: "quando mi vede" dobj(vede, mi); "hanno vinto la lotteria" dobj(vinto, lotteria)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Proper Noun Modifier

Counts the number of proper noun modifiers in the text. Gives an absolute number.

Ex: "Woody Allen" nnp(Allen, Woody); "Mariateresa Di Lascia" nnp(Lascia, Mariateresa) nnp(Lascia, Di)

Tint ha dato per "Mariateresa Di Lascia." :

Dependency Parse (enhanced plus plus dependencies): root(ROOT-0, Mariateresa-1) name(Mariateresa-1, Di-2) name(Mariateresa-1, Lascia-3) punct(Mariateresa-1, .-4)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Prepositional Phrases per Sentence

Calculates the syntactic complexity of the text. Calculates the prepositional phrases per sentence.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of prepositional phrases / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_PP_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure) of lexical words (LW).

Formula:

```
for each lexical word token:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of lexical word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Flesch Reading Ease

Calculates the Flesch Reading Ease measure of readability.

Formula:

```
206.835 - (84.6 * mean_token_length_in_syllables) - (1.015 *
mean_sentence_length_in_tokens)
```

Scores can be interpreted as shown in the table below:

100.00–90.00	5th grade	Very easy to read. Easily understood by an average 11-year-old student.
90.0–80.0	6th grade	Easy to read. Conversational English for consumers.
80.0–70.0	7th grade	Fairly easy to read.
70.0–60.0	8th and 9th grade	Plain English. Easily understood by 13- to 15-year-old students.
60.0–50.0	10th to 12th grade	Fairly difficult to read.
50.0–30.0	College	Difficult to read.
30.0–0.0	College graduate	Very difficult to read. Best understood by university graduates.

This feature was originally defined and optimised for English. CTAP allows to apply it also to German and Italian, but it's up to the user to decide to what extent the results are reliable.

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
```

Syntactic Complexity Feature: Mean Length of Verb Cluster

Calculates the syntactic complexity of the text. T Calculates the mean length of verb cluster.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of words / number of verb clusters
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
ParseTreeAnnotator.xml
```

```
NSyntacticConstituent_VC_Feature.xml
```

Lexical Sophistication Feature: Imageability (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Imageability norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Hierarchy Connectives

Calculates the number of hierarchy connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The lists contains 36 connectives.

Example: a questo punto anche anteriormente anzitutto appresso come di seguito dopo in aggiunta in conclusione

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Syntactic Complexity Feature: Sentence Complexity Ratio

Calculates the syntactic complexity of the text. Calculates the sentence complexity ratio.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of clauses / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_C_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency Per Million Words (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure per million words) of functional words (FW).

Formula:

```
for each functional word token:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words in the
Google corpus ( number of lines in the file ) / 1000 000 )

sum of LFs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Imageability (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Imageability norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional word tokens of the text found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Familiarity Per Million Words (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives

FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Familiarity Per Million Words) of functional words (FW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each functional word type:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word in the subtlex-[LANG].csv file (this word’s
FREQCOUNT added to the sum)

sum of FAs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (no modifier weight)

Calculates the average frequency of high adjacent IC (according to the DLT with cancelled modifier weight configuration) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
```

```
POSAnnotator.xml
DependencyParseAnnotator.xml
```

POS Density Feature: Emoticon

Calculates emoticon density of the text. Emoticons include for Italian: XE.

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
emoticonDensity = numEmoticons / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Imageability (All Lemmas)

Calculates lexical sophistication of the text.

A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

All lemmas features take into consideration all lemmas, while unique lemmas features calculate only unique lemmas.

This feature calculates lexical sophistication with the Imageability norm list from the Burani et al., 2001, containing 626 Italian nouns. https://www.istc.cnr.it/en/grouppage/varless

Formula:

```
sum of Imageability values for lemmas of the text found in Burani's list /
number of lemmas of the text found in Burani's list
```

```
Words of the text that are not in Burani's list are ignored.
```

Bib. ref.: Cristina Burani, Lisa S Arduino, Laura Barca, "Una base di dati sui valori di età di acquisizione, frequenza, familiarità, immaginabilità, concretezza, e altre variabili lessicali e sublessicali per 626 nomi dell'italiano", Giornale Italiano di Psicologia, January 2001

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Frequency (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list of lexical words (LW).

Formula:

```
sum of FREQCOUNTs of lexical word types / number of lexical word types
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: Total IC at Finite Verb (less coordination weight + higher verb weight)

Calculates the total integration cost (according to the DLT with less coordination weight and increased verb weight) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Lexical Sophistication Feature: Kucera and Francis (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Kucera and Francis norm list of frequencies (1967), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional word tokens of the text found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Concreteness (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes

a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Concreteness norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Adjectival Modifiers per Sentence

Calculates the syntactic complexity of the text. Calculates the number of adjectival modifiers per sentence.

An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP Used also for numbers, when they represent age. John Smith (33) ...

Ex: "nuovo record" amod(record, nuovo) "nell'ultima edizione della famosa maratona" amod(maratona, famosa)

Formula:

```
number of adjectival modifiers / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AdjectivalModifier_Feature.xml
```

```
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Google Books Word Familiarity Per Million Words (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Familiarity Per Million Words) of lexical words (LW).

Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each lexical word token:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word (this word’s FREQCOUNT added to the sum).

sum of FAs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Conjunctive Present Verb

Calculates conjunctive present verbs density of the text. Conjunctive present verbs include for Italian: Vcp Vcp3 VAcp VAcp3 VMcp VMcp3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
conjunctivePresentVerbDensity = numConjunctivePresentVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Contextual Diversity (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Contextual Diversity measure) of lexical words (LW).

Formula:

```
for each word type:
        CD = number of films in which the word appears

sum of CDs / number of word types
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Modal

Calculates modal verb density of the text. Modal verbs include for English: the Penn Treebank tags MD, for German: the Tiger tags VMFIN VMINF VMPP, for Italian: VMip VMip3 VMii VMii3 VMis VMis3 VMif VMif3 VMcp VMcp3 VMci VMci3 VMdp VMdp3 VMg VMp VMf VMm

Formula:

```
modalDensity = numModal / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Frequency (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list of all words (AW).

Formula:

```
sum of FREQCOUNTs of word tokens / number of word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Attributives per Sentence

Calculates the syntactic complexity of the text. Calculates the number of attributives per sentence.

An attributive is a complement of a copular verb such as "to be", "to seem", "to appear". Currently, the converter only recognizes WHNP complements.

benché Allende fosse già presidente attr(fosse, presidente) : Tint ha messo cop(fosse, presidente) "è la verità" attr(è, verità)

Formula:

```
number of attributives / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_Attributive_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Syntactic Complexity Feature: Prepositional Phrases per T-Unit

Calculates the syntactic complexity of the text. Calculates the prepositional phrases per T-Unit.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of prepositional phrases / number of T-Units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_PP_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency Per Million Words (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure per million words) of lexical words (LW).

Formula:

```
for each lexical word token:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the
Google corpus ( number of lines in the file ) / 1000 000 )

sum of LFs / number of lexical word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Mean Parse Tree Depth Feature

Calculates the mean parse tree depth of the text.

Formula:

```
sum of parse tree depths of all the sentences of the text / number of sentences
of the text
```

Note that different syntactic parsers are used for different languages: - Stanford dependencies are created for English and German - UD dependencies are created for Italian.

UD dependencies are "flatter" than Stanford dependencies. For that reason the mean parse tree depth for Italian tends to be smaller than the mean parse tree depth for English or German.

Mean parse tree depth has to be compared for different texts written in the same language.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Noun Phrase

Counts the number of noun phrases in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Tokens

Calculates the number of tokens in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency Per Million Words (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure per million words) of all words (AW):

Formula:

```
for each token:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the open subtitles corpus ( number of lines in the SUBTLEX-[lang].csv file ) / 1000 000 )

sum of LFs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Adjectival Complement

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of adjectival complements in the text. Gives an absolute number.

An adjectival complement of a verb is an adjectival phrase which functions as the complement (like an object of the verb)

Ex: "Lo hanno dichiarato colpevole" acomp(dichiarato, colpevole) : Tint a dato 'amod' in questo caso. "Considero Maria simpatica" acomp(considero, simpatica)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Lexical Words

Calculates lexical word density of the text. Lexical words include for English: the Penn Treebank tags JJ, JJR, JJS, RB, RBR, RBS, WRB, VB, VBD, VBG, VBN, VBP, VBZ; for German: the Tiger tags: ADJA ADJD ADV NN NE VVFIN VVIMP VVINF VVIZU VVPP VMFIN VMIMP VMINF VMIZU VMPP FM XY for Italian: the tags: As,Ap,An,APs,APp,APn,SP,S,Ss,Sp,Sn, SW, SWs,SWp,SWn,B,BB,Vip,Vip3,Vii,Vii3,Vis,Vis3,Vif,Vif3,Vcp,Vcp3,Vci,Vci3,Vdp,Vdp3,Vg,Vp,Vf,Vm

Formula:

```
lexicalWordDensity = numLexicalWords / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Argumentative Connectives per Token

Calculates the cohesive complexity of the text. Calculates the argumentative connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian. The lists contains 5 connectives: a proposito di in relazione a per quanto riguarda relativamente a riguardo a

Formula:

```
number of argumentative connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Lexical Sophistication Feature: Meaningfulness Pavio (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Pavio norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical
word tokens of the text found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Demonstrative Pronoun

Calculates Demonstrative Pronoun density of the text. Demonstrative Pronouns include for German: the Tiger tags PDS PDAT. for Italian: PD.

Availability:

```
This feature is NOT available for: English.
```

Formula:

```
demPronDensity = numDemPron / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Contextual Diversity (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Logarithmic Contextual Diversity measure) of all words (AW).

Formula:

```
for each word type:
        CD = log10 (number of films in which the word appears + 1)

sum of CDs / number of word types
```

Availability:

```
This feature is NOT available for German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Relative Clauses per T-Unit

Calculates the syntactic complexity of the text. Calculates the relative clauses per T-Unit.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of relative clauses phrases / number of T-Units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_RS_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Lexical Sophistication Feature: Minimal Age of Active Use in KCT (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Minimal Age of Active Use in KCT list of all words (AW).

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: Total IC at Finite Verb (higher verb weight)

Calculates the total integration cost (according to the DLT with its additional verb weight configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Number of syllables

Calculates the number of syllables in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (no modifier weight + less coordination weight)

Calculates the average maximal integration cost per clause (according to the DLT with cancelled modifier weight and reduced coordination costs configuration) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

SentenceAnnotator.xml TokenAnnotator.xml LemmaAnnotator.xml MorphologicalTagAnnotator.xml POSAnnotator.xml DependencyParseAnnotator.xml

Lexical Sophistication Feature: Google Books Word Frequency (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list of lexical words (LW).

Formula:

```
sum of FREQCOUNTs of lexical word tokens / number of lexical word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

```
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency Per Million Words (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure per million words) of all words (AW).

Formula:

```
for each token:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words in the
Google corpus ( number of lines in the file ) / 1000 000 )

sum of LFs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Temporal Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the temporal connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The Italian lists contains 55 connectives.

Example: a questo punto alla fine allora allorché allorquando anni fa antecedente a

The German lists contains 26 connectives.

Example: dann : selbst dann wenn,auch dann wenn,sogar dann wenn,wenn-dann bevor : bevor nicht zuletzt : nicht zuletzt anfangs bald bereits

Formula:

```
number of temporal connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
NConnectives_Breindl_Temporal_Feature.xml
```

Number of Connectives: Breindl Single-Word Connectives

Calculates the number of all single-word connectives listed by Breindl. Gives an absolute number.

The lists contains 60 single-word connectives.

Example: aber allerdings also anfangs außerdem bald

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Number of Connectives: Concessive Connectives (Breindl for German)

Calculates the number of concessive connectives listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The German list contains 6 connectives: allerdings dennoch obwohl sowieso trotzdem wenngleich

The Italian lists contains 7 connectives: ancorché ancorquando benché malgrado quantunque sebbene seppure

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Pavio (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Pavio norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional word tokens of the text found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Age of Acquisition norm list by Gilhooly and Logie (1980), which is included in the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of AoA values for tokens found in the norm list / number of tokens of the
text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Possessive Ending

Calculates possessive ending density of the text. Possessive endings include for English the Penn Treebank tag POS. ### Availability: This feature is NOT available for: German, Italian

Formula:

```
POSDensity = numPOSs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

SD Sentence Length in Tokens

Calculates the standard deviation of sentence length in number of tokens.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
LetterAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (Unique Lemmas)

Calculates lexical sophistication of the text.

A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

All lemmas features take into consideration all lemmas, while unique lemmas features calculate only unique lemmas.

This feature calculates lexical sophistication with the Imageability norm list from the Burani et al., 2001, containing 626 Italian nouns. https://www.istc.cnr.it/en/grouppage/varless

Formula:

```
sum of AoA values for lemmas of the text found in Burani's list / number of
lemmas of the text found in Burani's list

Words of the text that are not in Burani's list are ignored.
```

Bib. ref.: Cristina Burani, Lisa S Arduino, Laura Barca, "Una base di dati sui valori di età di acquisizione, frequenza, familiarità, immaginabilità, concretezza, e altre variabili lessicali e sublessicali per 626 nomi dell'italiano", Giornale Italiano di Psicologia, January 2001

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
```

Number of Syntactic Constituents: Relative Clauses

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of relative clauses in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Adverbial Clauses per Sentence

Calculates the syntactic complexity of the text. Calculates the number of adverbial modifiers per sentence.

An adverbial clause modifier of a VP or S is a clause modifying the verb (temporal clause, consequence, conditional clause, purpose clause, etc.)

"Quando venne coniato il termine, esso era applicato a un particolare fenomeno atmosferico" advcl(applicato, coniato) "Se non entrano in contatto con goccioline d'acqua, questi gas possono depositarsi sul suolo" advcl(depositarsi, entrano)

Formula:

```
number of adverbial clause modifiers / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AdverbialClause_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Syntactic Complexity Feature: Adverbial Modifiers per Sentence

Calculates the syntactic complexity of the text. Calculates the number of Adverbial modifiers per sentence.

An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP Used also for numbers, when they represent age. John Smith (33) ...

Ex: "nuovo record" amod(record, nuovo) "nell'ultima edizione della famosa maratona" amod(maratona, famosa)

Formula:

```
number of adverbial modifiers / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AdverbialModifier_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Concreteness (Unique Lemmas)

Calculates lexical sophistication of the text.

A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

All lemmas features take into consideration all lemmas, while unique lemmas features calculate only unique lemmas.

This feature calculates lexical sophistication with the Concreteness norm list from the Burani et al., 2001, containing 626 Italian nouns. https://www.istc.cnr.it/en/grouppage/varless

Formula:

```
sum of Concreteness values for lemmas of the text found in Burani's list /
number of lemmas of the text found in Burani's list

Words of the text that are not in Burani's list are ignored.
```

Bib. ref.: Cristina Burani, Lisa S Arduino, Laura Barca, "Una base di dati sui valori di età di acquisizione, frequenza, familiarità, immaginabilità, concretezza, e altre variabili lessicali e sublessicali per 626 nomi dell'italiano", Giornale Italiano di Psicologia, January 2001

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
```

Number of Syntactic Constituents: Complex Noun Phrase

Counts the number of complex noun phrases in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Mean Length of Complex T-unit

Calculates the syntactic complexity of the text. Calculates the mean length of complex T-unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of words / number of complex T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CT_Feature.xml
```

Number of Connectives: Breindl Other Connectives

Calculates the number of unspecified connectives connectives listed by Breindl. Gives an absolute number.

The lists contains 10 connectives: jedoch sowie falls hierdurch höchstens oder ohnehin sofern sonst sprich

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Familiarity Per Million Words (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Familiarity Per Million Words) of all words (AW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each token:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word in the subtlex-[LANG].csv file (this word’s
FREQCOUNT added to the sum)

sum of FAs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Concessive Connectives (Eisenberg for German)

Calculates the number of concessive connectives for German listed by Eisenberg. Gives an absolute number.

The list contains 38 connectives.

Example: trotz : trotz allem unbeschadet abgesehen von ungeachtet obwohl obgleich

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

POS Density Feature: Plural Proper Noun

Calculates plural proper noun density of the text. Plural proper nouns include for English the Penn Treebank tag NNPS.

Availability:

```
This feature is NOT available for: German, Italian
```

Formula:

```
NNPSDensity = numNNPS / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Predeterminer

Counts the number of predeterminers in the text. Gives an absolute number.

A predeterminer is the relation between the head of an NP and a word that precedes and modifies the meaning of the NP determiner.

Ex: "Tutte le piccole e medie aziende" predet(aziende, tutte) Tint ha dato 'det:predet(aziende-6, Tutte-1)'

The feature is not available for English, German.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Frequency (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list of all words (AW).

Formula:

```
sum of FREQCOUNTs of word types / number of word types
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Familiarity Per Million Words (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Familiarity Per Million Words) of all words (AW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each token:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word (this word’s FREQCOUNT added to the sum).

sum of FAs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Thorndike Lorge (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes

a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Thorndike-Lorge norm list of frequencies (1944), which is included in the MRC Psycholinguistic Database (http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html).

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical
word tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (no modifier weight + less coordination weight + added verb weight)

Calculates the average maximal integration cost per clause (according to the DLT with cancelled modifier weight, reduced coordination weight, and increased verb weight) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Lexical Sophistication Feature: Imageability (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Imageability norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Complex Prepositional Phrase

Counts the number of complex prepositional phrases in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Complex Nominals per Sentence

Calculates the syntactic complexity of the text. Calculates the complex nominals per sentence.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of complex nominals / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CN_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Google Books Word Familiarity Per Million Words (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Familiarity Per Million Words) of functional words (FW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each functional word token:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word (this word’s FREQCOUNT added to the sum).

sum of FAs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Main Verb

Calculates Main Verb density of the text. Main Verbs include for German: the Tiger tags VVFIN VVIMP VVINF VVIZU VVPP. for Italian: Vip,Vip3,Vii,Vii3,Vis,Vis3,Vif,Vif3,Vcp,Vcp3,Vci,Vci3,Vdp,Vdp3,Vg,Vp,Vf,Vm

Availability:

```
This feature is NOT available for English.
```

Formula:

```
mainVerbDensity = numMainVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Frequency (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list of all words (AW).

Formula:

```
sum of FREQCOUNTs of word types / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: 3rd Person Singular Verb

Calculates 3rd person singular verb density of the text. Third person singular verbs include for English the Penn Treebank tag VBZ.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
VBZDensity = numVBZs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Colerado (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Colerado norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical
word tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Brown (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Brown norm list of frequencies by Brown (1984), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional words found in the norm list / number of functional
words of the text found in the norm list
```

```
Functional words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Complex Nominals per T-unit

Calculates the syntactic complexity of the text. Calculates the complex nominals per T-unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of complex nominals / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CN_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

POS Density Feature: Infinite Verb

Calculates Infinite Verb density of the text. Infinite Verbs include for German: the Tiger tags VVINF VAINF VMINF. for Italian: the tags VMf VAf Vf.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
infVerbDensity = numInfVerb / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Article

Calculates article density of the text. Articles for English: include the Penn Treebank tag DT, for German: the Tiger tag ART, for Italian: RD, RI

Formula:

```
DTDensity = numDT / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Temporal Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of temporal modifiers in the text. Gives an absolute number.

A temporal modifier (of a VP, NP, or an ADJP is a bare noun phrase constituent that serves to modify the meaning of the constituent by specifying a time. (Other temporal modifiers are prepositional phrases and are introduced as prep.)

Ex: "L'allarme è scattato la scorsa settimana" tmod(scattato, settimana) Tint ha dato: nmod(scattato-4, settimana-7)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Contextual Diversity (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Contextual Diversity measure) of all words (AW).

Formula:

```
for each word type:
        CD = number of films in which the word appears

sum of CDs / number of word types
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Other Connectives per Token (Eisenberg for German)

Calculates the cohesive complexity of the text. Calculates the other connectives listed by Eisenberg per token: connectives that are not included in lists with defined categories. The list contains 10 connectives: jedoch sowie falls hierdurch höchstens oder ohnehin sofern sonst sprich

Formula:

```
number of other connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Eisenberg_Other_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure) of lexical words (LW).

Formula:

```
for each lexical word type:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of lexical word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Complex Prepositional Phrases per T-Unit

Calculates the syntactic complexity of the text. Calculates the complex prepositional phrases per T-Unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of complex prepositional phrases / number of T-Units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CPP_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Lexical Sophistication: Percentage of Lemmas Listed in the De Mauro Basic Dictionary (all lemmas)

Calculates the percentage of lemmas listed in the De Mauro basic dictionary.

Formula:

```
( number of lemmas listed in the De Mauro dictionary / number of lemmas
(excluding punctuations) ) * 100
```

Bib. ref.: Il Nuovo vocabolario di base della lingua italiana. A cura di Tullio De Mauro. 23 dicembre 2016. https://dizionario.internazionale.it/nuovovocabolariodibase

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
LemmaAnnotator.xml
```

Number of Connectives: Adversative or Concessive Connectives (Eisenberg for German)

Calculates the number of adversative or concessive connectives for German listed by Eisenberg. Gives an absolute number.

The lists contains 78 connectives.

Example: gegen entgegen zuwider statt : statt dass

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency Per Million Words (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure per million words) of all words (AW).

Formula:

```
for each word type:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words in the
Google corpus ( number of lines in the file ) / 1000 000 )
```

```
sum of LFs/ number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Concreteness (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Concreteness norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional words found in the norm list / number of functional
words of the text found in the norm list
```

```
Functional words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Possessive WH Pronoun

Calculates possesive WH pronoun density of the text. Possesive WH pronouns include for English the Penn Treebank tag WP$.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
possessive WH pronouns density = number of possessive WH pronouns / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Indicative Imperfect Tense Verb

Calculates indicative imperfect tense verb density of the text. Indicative imperfect tense verbs include for Italian: Vii Vii3 VAii Vii3 VMii VMii3.

Availability:

```
This feature is NOT available for: English, German
```

Formula:

```
VBDDensity = numVBDs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Colerado (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Colerado norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Conjunction

Calculates conjunction density of the text. Conjunctions include for English: the Penn Treebank tags CC, and IN, for German: the Tiger tags KOUI KOUS KON KOKOM, for Italian: tags CC CS.

Formula:

```
conjDensity = numConj / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Imageability (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Imageability norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Temporal Connectives (Eisenberg for German)

Calculates the number of temporal connectives for German listed by Eisenberg. Gives an absolute number.

The German list contains 71 connectives.

Example: nach nachdem kaum wonach worauf dann : selbst dann wenn,auch dann wenn,sogar dann wenn,wenn-dann

The Italian list contains 55 connectives.

Example: a quei tempi adesso alla fine allora allorché allorquando anni fa

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Number of Syntactic Constituents: Dependent Clauses

Calculates the number of a specific syntactic constituents in the text. Counts the number of dependent clauses in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Coordinations per Sentence

Calculates the syntactic complexity of the text. Calculates the number of coordinations per sentence.

Formula:

```
number of coordinations / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_Coordination_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Cohesive Complexity Feature: Multifunctional Connectives per Token

Calculates the cohesive complexity of the text. Calculates the multifunctional connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Multifunctional connectives appear in more than 1 list. The lists contains 23 connectives.

Example: The connectives 'dopo' and 'poi' both have 2 meanings, temporal and hierarchical. So they appear in 3 lists: temporal, hierarchical and multifunctional.

Formula:

```
number of multifunctional connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Lexical Sophistication Feature: SUBTLEX Word Familiarity Per Million Words (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Familiarity Per Million Words) of lexical words (LW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each lexical word type:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word in the subtlex-[LANG].csv file (this word’s
FREQCOUNT added to the sum)

sum of FAs / number of lexical word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Hypothetical and Conditional Connectives per Token

Calculates the cohesive complexity of the text. Calculates the hypothetical and conditional connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian.

The list contains 20 connectives.

Example: ammettendo che nel caso in cui partendo dal presupposto che purché qualora

Formula:

```
number of hypothetical and conditional connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Number of Connectives: Causal Connectives (Eisenberg for German)

Calculates the number of causal connectives for German listed by Eisenberg. Gives an absolute number.

The list contains 58 connectives.

Example: andernfalls ansonsten dann : selbst dann wenn,auch dann wenn,sogar dann wenn,wenn-dann unter Umständen eventuell wegen

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Number of Syntactic Constituents: Auxiliary Passive

Counts the number of passive auxiliaries in the text. Gives an absolute number.

A passive auxiliary of a clause is a non-main verb of the clause which contains the passive information

Ex: "Significa che sono state tagliate le vie nervose" auxpass(tagliate, state) "Le donne candidate vengono ritenute intelligenti" auxpass(ritenute, vengono)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

```
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Colerado (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Colerado norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Clauses

Counts the number of clauses in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Verb Phrases per T-unit

Calculates the syntactic complexity of the text. Calculates the verb phrases per T-unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of verb phrases / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_VP_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

POS Density Feature: Cardinal Number

Calculates cardinal number density of the text. Cardinal numbers include for English: the Penn Treebank tags CD, for German: the Tiger tag CARD, for Italian: N.

Formula:

```
CDDensity = numCDs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Symbol

Calculates symbol density of the text. Symbols include for English the Penn Treebank tag SYM, for Italian: XX XE.

Availability:

```
This feature is NOT available for: German
```

Formula:

```
SYMDensity = numSYMs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency Per Million Words (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure per million words) of functional words (FW).

Formula:

```
for each functional word type:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words in the
Google corpus ( number of lines in the file ) / 1000 000 )

sum of LFs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Informativeness Per Million Words (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives

FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Informativeness Per Million Words) of all words (AW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each token:
        WI = number of words with the same first 3 characters and of the same
length as this word in the subtlex-[LANG].csv file (this word included)

sum of WIs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Frequency (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list of functional words (FW).

Formula:

```
sum of FREQCOUNTs of functional word tokens / number of functional word tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Mean Age of Active Use in KCT (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Mean Age of Active Use in KCT list of lexical words (LW).

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Expletive

Counts the number of expletives in the text. Gives an absolute number.

This relation captures an existential \there". The main verb of the clause is the governor

Ex: "Se vi è un bersaglio" expl(è, vi); "Ci sarà un rinnovamento" expl(sarà, ci)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Adverbial Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of adverbial modifiers in the text. Gives an absolute number.

An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP Used also for numbers, when they represent age. John Smith (33) ...

Ex: "nuovo record" amod(record, nuovo) "nell'ultima edizione della famosa maratona" amod(maratona, famosa)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Possessive Pronoun

Calculates possessive pronoun density of the text. Possesive pronouns include for English: the Penn Treebank tag PRP$, for German: the Tiger tags PPOSS, PPOSAT, for Italian: PP.

Formula:

```
PRPSDensity = numPRPSs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Richness: MTLD Lemma

Calculates the MTLD measure of lexical diversity. Was translated into Java by Nadezda Okinina from the Python implementation of John Frens: https://github.com/jfrens/lexical_diversity

MTLD is a more sophisticated measure of lexical diversity compared to TTR (type token ratio). TTR being highly text length dependent, more sophisticated formulas were invented to overcome this shortcoming, MTLD being one of them. The efficiency of such new formulas is subject to debate.

If the text length is inferior to 50 tokens, gives the value -1.

Bib. ref.: McCarthy, P. M. and Jarvis, S. (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. Behavior Research Methods, 42(2):381–392.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
LemmaAnnotator.xml
NTokenFeature.xml
NLemmaFeature.xml
NTokenTypeFeature.xml
```

Lexical Sophistication Feature: SUBTLEX Word Informativeness Per Million Words (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Informativeness Per Million Words) of functional words (FW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each functional word type:
        WI = number of words with the same first 3 characters and of the same
length as this word in the subtlex-[LANG].csv file (this word included)

sum of WIs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: Total IC at Finite Verb (no modifier weight)

Calculates the total integration cost (according to the DLT with cancelled modifier weight configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

POS Density Feature: Comparative Adjective

Calculates comparative adjective density of the text. Comparative adjectives include for English the Penn Treebank tag JJR.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
JJRDensity = numJJR     / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Attributive

Counts the number of attributives in the text. Gives an absolute number.

An attributive is a complement of a copular verb such as "to be", "to seem", "to appear". Currently, the converter only recognizes WHNP complements.

Ex: benché Allende fosse già presidente attr(fosse, presidente) : Tint ha messo cop(fosse, presidente) "è la verità" attr(è, verità)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Kucera and Francis (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Kucera and Francis norm list of frequencies (1967), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Adverbial Clause Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of adverbial clause modifiers in the text. Gives an absolute number.

An adverbial clause modifier of a VP or S is a clause modifying the verb (temporal clause, consequence, conditional clause, purpose clause, etc.)

Ex: "Quando venne coniato il termine, esso era applicato a un particolare fenomeno atmosferico" advcl(applicato, coniato) "Se non entrano in contatto con goccioline d'acqua, questi gas possono depositarsi sul suolo" advcl(depositarsi, entrano)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Thorndike Lorge (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Thorndike-Lorge norm list of frequencies (1944), which is included in the MRC Psycholinguistic Database (http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html).

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Pavio (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Pavio norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Marker

Counts the number of markers in the text. Gives an absolute number.

A marker is the word introducing a finite clause subordinate to another clause. For a complement clause, this will typically be "that" or "whether". For an adverbial clause, the marker is typically a preposition like "while" or "although". The mark is a dependent of the subordinate clause head.

Ex: "Quando sono stati investiti dalla violenza del fulmine" mark(investiti, quando); Tint ha dato: 'advmod(investiti-4, Quando-1)' "Mentre leggono le notizie" mark(leggono, mentre)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

SD Token Length in Letters

Calculates the standard deviation of token length in number of letters.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
LetterAnnotator.xml
```

Lexical Richness: HDD Token

Calculates the HDD measure of lexical diversity. Was translated into Java by Nadezda Okinina from the Python implementation of John Frens: https://github.com/jfrens/lexical_diversity

HDD is a more sophisticated measure of lexical diversity compared to TTR (type token ratio). TTR being highly text length dependent, more sophisticated formulas were invented to overcome this shortcoming, HDD being one of them. The efficiency of such new formulas is subject to debate.

If the text length is inferior to 50 tokens, gives the value -1.

Bib. ref.: McCarthy, P. M. and Jarvis, S. (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. Behavior Research Methods, 42(2):381–392.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

POS Density Feature: Non 3rd Person Singular Verb

Calculates non 3rd person singular verb density of the text. Non-third person singular verbs include for English the Penn Treebank tag VBP.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
VBPDensity = numVBPs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Additive Connectives per Token (Eisenberg for German)

Calculates the cohesive complexity of the text. Calculates the additive connectives per token, listed by Eisenberg for German. The list contains 41 connectives.

Example: einschließlich samt nebst inklusive zuzüglich ohne zu

Formula:

```
number of additive connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
NConnectives_Breindl_Additive_Feature.xml
```

Number of Connectives: Adversative or Concessive Connectives (Breindl for German)

Calculates the number of adversative or concessive connectives listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The German list contains 17 connectives.

Example: zwar : zwar-aber,zwar-doch aber sondern : sondern-auch nur : nicht-nur

The Italian lists contains 38 connectives.

Example: al contrario all'inverso all'opposto anzi anziché cionondimeno

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Syntactic Complexity Feature: Verb Cluster per Sentence

Calculates the syntactic complexity of the text. Calculates the verb cluster per sentence.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of verb cluster / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_VC_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Number of Syntactic Constituents: Auxiliary

Counts the number of auxiliaries in the text. Gives an absolute number.

An auxiliary of a clause is a non-main verb of the clause, e.g., a modal auxiliary, or a form of "be", "do" or "have" in a periphrastic tense

Ex: "quella che ha fatto" aux(fatto, ha) "era convinto di dover vivere" aux(vivere, dover)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Variation Feature: Squared Verb Variation 1

Calculates lexical variation of the text. Lexical words are certain types of verbs, nouns, adjectives, and adverb. This feature calculates Verb variation.

Formula:

```
SVV1 = nVerbType^2 / nVerbToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (higher verb weight)

Calculates the average frequency of high adjacent IC (according to the DLT with its additional verb weight configuration) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Informativeness Per Million Words (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives

FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Informativeness Per Million Words) of all words (AW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each word type:
        WI = number of words with the same first 3 characters and of the same length as this word (this word included)

sum of WIs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Proportion of Tokens with 2 or more Syllables

Calculates percentage of word tokens with 2 or more syllables.

Formula:

```
number of tokens with 2 or more syllables / total number of tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
```

Mean Token Length in Syllables

Calculates the mean token length in syllables.

Formula:

```
sum of the lengths of all the tokens of the text / number of tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
LetterAnnotator.xml
NLetterFeature.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (less coordination weight)

Calculates the average maximal integration cost per clause (according to the DLT with reduced coordination weight configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib. ref.: Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of linguistic complexity. In: Image, language, brain, pp. 95–126.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Lexical Sophistication Feature: Imageability (Unique Lemmas)

Calculates lexical sophistication of the text.

A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

All lemmas features take into consideration all lemmas, while unique lemmas features calculate only unique lemmas.

This feature calculates lexical sophistication with the Imageability norm list from the Burani et al., 2001, containing 626 Italian nouns. https://www.istc.cnr.it/en/grouppage/varless

Formula:

```
sum of Imageability values for lemmas of the text found in Burani's list /
number of lemmas of the text found in Burani's list
```

```
Words of the text that are not in Burani's list are ignored.
```

Bib. ref.: Cristina Burani, Lisa S Arduino, Laura Barca, "Una base di dati sui valori di età di acquisizione, frequenza, familiarità, immaginabilità, concretezza, e altre variabili lessicali e sublessicali per 626 nomi dell'italiano", Giornale Italiano di Psicologia, January 2001

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
```

Syntactic Complexity Feature: Postnominal Modifier per Complex Noun Phrase

Calculates the syntactic complexity of the text. Calculates the postnominal modifier per complex noun phrase.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of postnominal modifier / number of complex noun phrases
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_PostN_Feature.xml
NSyntacticConstituent_CN_Feature.xml
```

Lexical Sophistication Feature: SUBTLEX Word Informativeness Per Million Words (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Informativeness Per Million Words) of lexical words (LW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each lexical word token:
        WI = number of words with the same first 3 characters and of the same
length as this word in the subtlex-[LANG].csv file (this word included)

sum of WIs / number of lexical word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Causal Connectives per Token (Eisenberg for German)

Calculates the cohesive complexity of the text. Calculates the causal connectives per token, listed by Eisenberg for German. The list contains 58 connectives.

Example: andernfalls ansonsten dann : selbst dann wenn,auch dann wenn,sogar dann wenn,wenn-dann unter Umständen eventuell wegen

Formula:

```
number of temporal connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Causal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Lexical Richness: Type Token Ratio (Corrected TTR)

Calculates the type token ratio of a text. A word type is a non-duplicated token.

Formula:

```
This features calculates the corrected TTR with the ### Formula:
CTTR = T/sqrt(2*N)

T stands for number of word types,
N stands for number of tokens.
```

Bib. ref.: Carroll, J. B. 1964. Language and Thought. Englewood Cliffs, NJ: Prentice-Hall.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

Lexical Sophistication Feature: Google Books Word Informativeness Per Million Words (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Informativeness Per Million Words) of functional words (FW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each functional word token:
        WI = number of words with the same first 3 characters and of the same
length as this word (this word included)

sum of WIs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Sentence Coordination Ratio

Calculates the syntactic complexity of the text. Calculates the sentence coordination ratio.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of clauses / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_T_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Cohesive Complexity Feature: Hierarchy Connectives per Token

Calculates the cohesive complexity of the text. Calculates the hierarchy connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian. The lists contains 36 connectives.

Example: a questo punto anche anteriormente anzitutto appresso come di seguito dopo in aggiunta in conclusione

Formula:

```
number of hierarchy connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Cohesive Complexity Feature: Causal Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the causal connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The Italian list contains 16 connectives.

Example: ché considerato che dacché dal momento che dappoiché dato che

The German list contains 8 connectives: denn : geschweige denn also deshalb deswegen folglich nämlich somit weil

Formula:

```
number of temporal connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Causal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Syntactic Complexity Feature: Complex T-unit Ratio

Calculates the syntactic complexity of the text. Calculates the complex T-unit ratio.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of complex T-units / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CT_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Number of Syntactic Constituents: Numeric Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of numeric modifiers in the text. Gives an absolute number.

A numeric modifier of a noun is any number phrase that serves to modify the meaning of the noun

Ex: "Ha cambiato due ministri" num(ministri, due); "Dopo cinque anni di lavoro" num(anni, cinque)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Mean Sentence Length in Letters

Calculates the mean sentence length in letters.

Formula:

```
sum of the lengths of all the sentences of the text / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
LetterAnnotator.xml
NLetterFeature.xml
```

POS Density Feature: Singular Proper Noun

Calculates singular proper noun density of the text. Singular proper nouns include for English: the Penn Treebank tag NNP, for German: NE

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
NNPDensity = numNNP / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Clausal Passive Subject

Counts the number of clausal passive subjects in the text. Gives an absolute number.

A clausal passive subject is a clausal syntactic subject of a passive clause. In the example below, "that she lied" is the subject.

"Che mentisse era sospettato da tutti" csubjpass(sospettato, mentisse): Tint ha dato 'nsubjpass(sospettato-4, mentisse-2)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Familiarity Per Million Words (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Familiarity Per Million Words) of all words (AW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each word type:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word in the subtlex-[LANG].csv file (this word’s
FREQCOUNT added to the sum)

sum of FAs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Coordinate Phrases per Sentence

Calculates the syntactic complexity of the text. Calculates the coordinate phrases per sentence.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of coordinate phrases / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CP_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Richness: MTLD Token

Calculates the MTLD measure of lexical diversity. Was translated into Java by Nadezda Okinina from the Python implementation of John Frens: https://github.com/jfrens/lexical_diversity

MTLD is a more sophisticated measure of lexical diversity compared to TTR (type token ratio). TTR being highly text length dependent, more sophisticated formulas were invented to overcome this shortcoming, MTLD being one of them. The efficiency of such new formulas is subject to debate.

If the text length is inferior to 50 tokens, gives the value -1.

Bib. ref.: McCarthy, P. M. and Jarvis, S. (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. Behavior Research Methods, 42(2):381–392.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

Cohesive Complexity Feature: Temporal Connectives per Token (Eisenberg for German)

Calculates the cohesive complexity of the text. Calculates the temporal connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The German list contains 71 connectives.

Example: nach nachdem kaum wonach worauf dann : selbst dann wenn,auch dann wenn,sogar dann wenn,wenn-dann

The Italian list contains 55 connectives.

Example: a quei tempi adesso alla fine allora allorché allorquando anni fa

Formula:

```
number of temporal connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
NConnectives_Eisenberg_Temporal_Feature.xml
```

POS Density Feature: Personal Pronoun

Calculates Personal Pronoun density of the text. Personal Pronouns include for English: the Penn Treebank tag PRP, for German: the Tiger tags PPER PRF, for Italian: PE.

Formula:

```
persPronDensity = numPersPron / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Singular Adjective

Calculates adjective density of the text. Plural adjectives include for Italian: As APs

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
adjDensity = numSingularAdjectives / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Richness: HDD Lemma

Calculates the HDD measure of lexical diversity. Was translated into Java by Nadezda Okinina from the Python implementation of John Frens: https://github.com/jfrens/lexical_diversity

HDD is a more sophisticated measure of lexical diversity compared to TTR (type token ratio). TTR being highly text length dependent, more sophisticated formulas were invented to overcome this shortcoming, HDD being one of them. The efficiency of such new formulas is subject to debate.

If the text length is inferior to 50 tokens, gives the value -1.

Bib. ref.: McCarthy, P. M. and Jarvis, S. (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. Behavior Research Methods, 42(2):381–392.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
LemmaAnnotator.xml
NTokenFeature.xml
NLemmaFeature.xml
NTokenTypeFeature.xml
```

Syntactic Complexity Feature: Noun Phrases per Sentence

Calculates the syntactic complexity of the text. Calculates the noun phrases per sentence.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of noun phrases / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_NP_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Number of Syntactic Constituents: Case

Counts the number of cases in the text. Gives an absolute number.

"programmi per il trattamento della pelle"

Tint ha dato:

Dependency Parse (enhanced plus plus dependencies): root(ROOT-0, programmi-1) case(trattamento-4, per-2) det(trattamento-4, il-3) nmod:per(programmi-1, trattamento-4) case(pelle-6, della-5) nmod:della(trattamento-4, pelle-6) punct(programmi-1, .-7)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure) of lexical words (LW).

Formula:

```
for each lexical word type:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of lexical word types
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Relative Clauses per Sentence

Calculates the syntactic complexity of the text. Calculates the relative clauses per sentence.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of relative clauses / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_RS_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (less coordination weight + higher verb weight)

Calculates the average maximal integration cost per clause (according to the DLT with less coordination weight and increased verb weight) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Syntactic Complexity Feature: Dependent Clauses per Sentence

Calculates the syntactic complexity of the text. Calculates the dependent clause per sentence.

Formula:

```
number of dependent clauses / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_DC_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Number of Syntactic Constituents: Multi-Word Expression

Counts the number of multi-word expressions in the text. Gives an absolute number.

the multi-word expression (modifier) relation is used for certain multi-word idioms that behave like a single function word. It is used for a closed set of dependencies between words in common multi-word expressions for which it seems difficult or unclear to assign any other relationships. At present, this relation is used inside the following expressions: rather than, as well as, instead of, such as, because of, instead of, in addition to, all but, such as, because of, instead of, due to. The boundaries of this class are unclear; it could grow or shrink a little over time

Ex: "top secret" mwe(top, secret); "meno che meno" mwe(meno, che) mwe(che, meno); "fino a prova contraria" mwe(fino, a)

Tint ha dato per la frase "Questo è un top secret fino a prova contraria." l'analisi seguente: Dependency Parse (enhanced plus plus dependencies): root(ROOT-0, secret-5) nsubj(secret-5, Questo-1) cop(secret-5, è-2) det(secret-5, un-3) amod(secret-5, top-4) case(prova-8, fino-6) mwe(fino-6, a-7) nmod:fino_a(secret-5, prova-8) amod(prova-8, contraria-9) punct(secret-5, .-10)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Adverb RB

Calculates adverb RB density of the text. Adverb RBs include for English the Penn Treebank tag RB.

Availability:

```
This feature is NOT available for: German, Italian
```

Formula:

```
RBDensity = numRBs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Noun Phrases per Clause

Calculates the syntactic complexity of the text. Calculates the noun phrases per clause.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of noun phrases / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_NP_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Number of Syntactic Constituents: Open Clausal Compliment

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of open clausal complements in the text. Gives an absolute number.

An open clausal complement (xcomp) of a VP or an ADJP is a clausal complement without its own subject, whose reference is determined by an external subject. These complements are always non-finite. The name xcomp is borrowed from Lexical-Functional Grammar

Ex: "Si dice che ami nuotare" xcomp(ami, nuotare) "Tutti gli esseri umani sanno di poter essere più di ciò che sono" xcomp(sanno, essere) Tint ha dato: 'xcomp(sanno-5, più-9)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Possession Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of possession modifiers in the text. Gives an absolute number.

The possession modifier relation holds between the head of an NP and its possessive determiner

Ex: "voce spese generali del suo studio" poss(studio, suo); Tint ha dato: det:poss(studio-6, suo-5)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Cohesive Complexity Feature: Consequence Connectives per Token

Calculates the cohesive complexity of the text. Calculates the consequence connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian. The list contains 14 connectives.

Example: così che da ciò si deduce che di conseguenza dunque

Formula:

```
number of consequence connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Lexical Sophistication Feature: Minimal Age of Active Use in KCT (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Minimal Age of Active Use in KCT list of functional words (FW).

Formula:

```
sum of values for functional word types found in the norm list / number of
functional word types of the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Indirect Object

Counts the number of indirect objects in the text. Gives an absolute number.

The indirect object of a VP is the noun phrase which is the (dative) object of the verb

Ex: "assegna alla proprietà anche una funzione sociale" iobj(assegna, proprietà); Tint ha dato: 'nmod:alla(assegna-1, proprietà-3)' "capacità riconosciuta- gli" iobj(riconosciuta-, gli); "ti ho dato l'oro" iobj(dato, ti)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Number

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of numbers in the text. Gives an absolute number.

An element of compound number is a part of a number phrase or currency amount

Ex: "35 milioni lordi a stagione" number(milioni, 35); "In città ci sono 500 mila persone" number(mila, 500) Tint ha dato: 'compound(mila-6, 500-5)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (original cost configuration)

Calculates the average maximal integration cost per clause (according to the DLT with its original cost configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Syntactic Complexity Feature: Verb Phrases per Sentence

Calculates the syntactic complexity of the text. Calculates the verb phrases per sentence.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of verb phrases / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_VP_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

POS Density Feature: Adverb

Calculates adverb density of the text. Adverbs include for English: the Penn Treebank tags RB, RBR, RBS, WRB. for German: the Tiger tag ADV. for Italian: B, BN

Formula:

```
advDensity = numAdverbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (less coordination weight + added verb weight)

Calculates the average frequency of high adjacent IC (according to the DLT with less coordination weight, and increased verb weight) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Syntactic Complexity Feature: Adjectival Complements per Sentence

Calculates the syntactic complexity of the text. Calculates the number of adjectival complements per sentence.

An adjectival complement of a verb is an adjectival phrase which functions as the complement (like an object of the verb)

"Lo hanno dichiarato colpevole" acomp(dichiarato, colpevole) : Tint a dato 'amod' in questo caso. "Considero Maria simpatica" acomp(considero, simpatica)

Formula:

```
number of adjectival complements / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AdjectivalComplement_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Mean Token Length in Letters

Calculates the mean token length in letters.

Formula:

```
sum of the lengths of all the tokens of the text / number of tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
LetterAnnotator.xml
NLetterFeature.xml
```

Number Of Letters

Count the number of letters in the document. Gives an absolute number.

AAE dependency: SentenceAnnotator.xml TokenAnnotator.xml LetterAnnotator.xml

POS Density Feature: Conjunctive Verb

Calculates conjunctive verbs density of the text. Conjunctive verbs include for Italian: Vcp Vcp3 Vci Vci3 VAcp VAcp3 VAci VAci3 VMcp VMcp3 VMci VMci3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
conjunctiveVerbDensity = numConjunctiveVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Minimal Age of Active Use in KCT (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Minimal Age of Active Use in KCT list of all words (AW).

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

SD Sentence Length in Syllables

Calculates the standard deviation of sentence length in number of syllables.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
LetterAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Frequency (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list of lexical words (LW).

Formula:

```
sum of FREQCOUNTs of lexical word tokens / number of lexical word tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Sentences

Counts the number of sentences in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Variation Feature: Modifier

Calculates lexical variation of the text. Lexical words are verbs, nouns, adjectives, and adverbs. This feature calculates Modifier variation. Modifiers are adjectives and adverbs.

Formula:

```
modifierVariation = nModifierType / nLexicalToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Past Tense Verb

Calculates past tense verb density of the text. Past tense verbs include for English the Penn Treebank tag VBD.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
VBDDensity = numVBDs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Nominal Subject

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of nominal subjects in the text. Gives an absolute number.

A nominal subject is a noun phrase which is the syntactic subject of a clause. The governor of this relation might not always be a verb: when the verb is a copular verb, the root of the clause is the complement of the copular verb, which can be an adjective or noun.

Ex: "è bastata una decisa accelerazione" nsubj(bastata, accelerazione); "Fabiana Luperini sta dominando la gara" nsubj(dominando, Luperini)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Base Form Verb

Calculates base form verb density of the text. Base form verbs include for English the Penn Treebank tag VB.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
VBDensity = numVBs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Determiner

Calculates determiner density of the text. Determiners include for English: the Penn Treebank tags PDT, DT and WDT, for German: the Tiger tags ART PDAT PIAT PPOSAT PRELAT PWAT, for Italian: DD, DE, DI, RD, RI

Formula:

```
detDensity = numDet / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Existential There

Calculates existential there density of the text. Existential there include for English the Penn Treebank tag EX.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
EXDensity = numEXs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Concreteness (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Concreteness norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional words found in the norm list / number of functional
words of the text found in the norm list
```

```
Functional words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Gerund Verb

Calculates gerund verb density of the text. Gerund verbs include for English the Penn Treebank tag VBG, for Italian: Vg VAg VMg.

Availability:

```
This feature is NOT available for: German
```

Formula:

```
VBGDensity = numVBGs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Concessive Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the concessive connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The German list contains 6 connectives: allerdings dennoch obwohl sowieso trotzdem wenngleich

The Italian lists contains 7 connectives: ancorché ancorquando benché malgrado quantunque sebbene seppure

Formula:

```
number of concessive connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Concessive_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Dependency Locality Theory: Total IC at Finite Verb (no modifier weight + less coordination weight)

Calculates the total integration cost (according to the DLT with cancelled modifier weight and reduced coordination costs configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

POS Density Feature: Possessive Adjective

Calculates adjective density of the text. Possessive adjectives include for Italian: APs APp APn

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
adjDensity = numPossessiveAdjectives / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Contextual Diversity (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Logarithmic Contextual Diversity measure) of functional words (FW).

Formula:

```
for each function word type:
        CD = log10 (number of films in which the word appears + 1)

sum of CDs / number of function word types
```

Availability:

```
This feature is NOT available for German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Concreteness (All Lemmas)

Calculates lexical sophistication of the text.

A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

All lemmas features take into consideration all lemmas, while unique lemmas features calculate only unique lemmas.

This feature calculates lexical sophistication with the Concreteness norm list from the Burani et al., 2001, containing 626 Italian nouns. https://www.istc.cnr.it/en/grouppage/varless

Formula:

```
sum of Concreteness values for lemmas of the text found in Burani's list /
number of lemmas of the text found in Burani's list
```

```
Words of the text that are not in Burani's list are ignored.
```

Bib. ref.: Cristina Burani, Lisa S Arduino, Laura Barca, "Una base di dati sui valori di età di acquisizione, frequenza, familiarità, immaginabilità, concretezza, e altre variabili lessicali e sublessicali per 626 nomi dell'italiano", Giornale Italiano di Psicologia, January 2001

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
```

POS Density Feature: To

Calculates To density of the text. Tos include for English the Penn Treebank tag TO, for German the Tiger tags VVIZU

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
TODensity = numTOs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Variation Feature: Adjective

Calculates lexical variation of the text. Lexical words are verbs, nouns, adjectives, and adverb. This feature calculates Adjective variation.

Formula:

```
adjectiveVariation = nAdjectiveType / nLexicalToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Verb Phrase

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of verb phrases in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Informativeness Per Million Words (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Informativeness Per Million Words) of all words (AW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each word type:
        WI = number of words with the same first 3 characters and of the same
length as this word in the subtlex-[LANG].csv file (this word included)

sum of WIs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Kucera and Francis (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Kucera and Francis norm list of frequencies (1967), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Age of Acquisition norm list by Gilhooly and Logie (1980), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of AoA values for lexical words found in the norm list / number of lexical
words of the text found in the norm list
```

```
Lexical words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Concreteness (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Concreteness norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical words found in the norm list / number of lexical words
of the text found in the norm list
```

```
Lexical words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Adversative Connectives (Eisenberg for German)

Calculates the number of adversative connectives for German listed by Eisenberg. Gives an absolute number.

The list contains 38 connectives.

Example: gegen entgegen zuwider statt : statt dass

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

POS Density Feature: Subordinating Conjunction

Calculates the density of Subordinating Conjunctions in the text. Subordinating Conjunctions include for German: the Tiger tags KOUS KOUI, for Italian: CS.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
subConjDensity = numSubordConj / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Noun Phrases per T-unit

Calculates the syntactic complexity of the text. Calculates the noun phrases per T-unit.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of noun phrases / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_NP_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

POS Density Feature: Predeterminer

Calculates predeterminer density of the text. Predeterminers include for English the Penn Treebank tag PDT, for Italian: T.

Availability:

```
This feature is NOT available for: German.
```

Formula:

```
PDTDensity = numPDT / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Clausal Subject

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of clausal subjects in the text. Gives an absolute number.

A clausal subject is a clausal syntactic subject of a clause, i.e., the subject is itself a clause. The governor of this relation might not always be a verb: when the verb is a copular verb, the root of the clause is the complement of the copular verb. In the two following examples, "what she said" is the subject.

Ex: "Che tu abbia ragione è indubbio" csubj(è, abbia) "A tutti è noto che il Costanzo decedette il giorno stesso" csubj(è, decedette). Tint ha dato: 'csubj(noto-4, decedette-8)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Thorndike Lorge (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Thorndike-Lorge norm list of frequencies (1944), which is included in the MRC Psycholinguistic Database (http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html).

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: T-units

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of T-units in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Interjection

Calculates interjection density of the text. Interjections include for English: the Penn Treebank tags UH for German: the Tiger tag ITJ for Italian: the tag I

Formula:

```
interDensity = numInter / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Proportion of Word Types with 2 or more Syllables

Calculates the percentage of words types with 2 or more syllables.

Formula:

```
number of word types with 2 or more syllables / total number of tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Informativeness Per Million Words (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Informativeness Per Million Words) of functional words (FW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each functional word type:
        WI = number of words with the same first 3 characters and of the same
length as this word (this word included)
```

```
sum of WIs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure) of all words (AW).

Formula:

```
for each token:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Interrogative Pronoun

Calculates Interrogative Pronoun density of the text. Interrogative Pronouns include for German: the Tiger tags PWS PWAT PWAV. for Italian: the tag PQ.

Availability:

```
This feature is NOT available for: English
```

Formula:

```
interPronDensity = numInterPron / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Variation Feature: Verb

Calculates lexical variation of the text. Lexical words are certain types of verbs, nouns, adjectives, and adverbs. This feature calculates Verb variation.

Formula:

```
verbVariation = nVerbType / nLexicalToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Adjective JJ

Calculates adjective JJ density of the text. Adjective JJs include for English the Penn Treebank tag JJ.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
JJDensity = numJJ / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Plural Noun

Calculates plural noun density of the text. Plural nouns include for English the Penn Treebank tag NNS, for Italian: Sp SWp.

Availability:

```
This feature is NOT available for: German
```

Formula:

```
NNSDensity = numNNS / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (higher verb weight)

Calculates the average maximal integration cost per clause (according to the DLT with its additional verb weight configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Syntactic Complexity Feature: Mean Length of Prepositional Phrase

Calculates the syntactic complexity of the text. T Calculates the mean length of prepositional phrase.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of word / number of prepositional phrases
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_PP_Feature.xml
```

Lexical Variation Feature: Lexical

Calculates lexical variation of the text. Lexical words are verbs, nouns, adjectives, and adverb.

Formula:

```
lexicalVariation = nLexicalType / nLexicalToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Contextual Diversity (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Logarithmic Contextual Diversity measure) of functional words (FW).

Formula:

```
for each function word token:
        CD = log10 (number of films in which the word appears + 1)

sum of CDs / number of function word tokens
```

Availability:

```
This feature is NOT available for German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Prepositional Phrases per Clause

Calculates the syntactic complexity of the text. Calculates the prepositional phrases per clause.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of prepositional phrases / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_PP_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Number of Syntactic Constituents: Coordination

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of coordinations in the text. Gives an absolute number.

A coordination is the relation between an element of a conjunct and the coordinating conjunction word of the conjunct. (Note: different dependency grammars have different treatments of coordination. We take one conjunct of a conjunction (normally the first) as the head of the conjunction.) A conjunction may also appear at the beginning of a sentence. This is also called a cc, and dependent on the root predicate of the sentence.

balliamo, suoniamo e cantiamo : 1 coordination balliamo, e suoniamo e cantiamo: 2 coordinations

"arriviste ed ambiziose" cc(arriviste, ed) "molto o abbastanza" cc(molto, o)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Variation Feature: Adverb

Calculates lexical variation of the text. Lexical words are verbs, nouns, adjectives, and adverb. This feature calculates Adverb variation.

Formula:

```
adverbVariation = nAdverbType / nLexicalToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Mean Length of Noun Phrase

Calculates the syntactic complexity of the text. Calculates the mean length of noun phrase.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of words / number of noun phrases
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_NP_Feature.xml
```

Lexical Sophistication Feature: Minimal Age of Active Use in KCT (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Minimal Age of Active Use in KCT list of lexical words (LW).

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical
word tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Adversative Connectives (Breindl for German)

Calculates the number of adversative connectives listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The German list contains 10 connective groups.

Example: zwar : zwar-aber,zwar-doch aber sondern : sondern-auch nur : nicht-nur einerseits

The Italian lists contains 38 connectives.

Example: al contrario all'inverso all'opposto anzi anziché ciononondimeno

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Dependency Locality Theory: Total IC at Finite Verb (original cost configuration)

Calculates the total integration cost (according to the DLT with its original cost configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

POS Density Feature: Noun

Calculates noun density of the text. Nouns include for English: the Penn Treebank tags NN, NNS, NNP and NNPS, for German: the Tiger tags NN NE, for Italian: the tags Ss Sp Sn SWs SWp SWn.

Formula:

```
nounDensity = numNouns / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Abbreviation

Calculates abbreviation density of the text. Abbreviations include for Italian: SA.

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
abbreviationDensity = numAbbreviations / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Dependency Locality Theory: Maximal IC at Finite Verb (no modifier weight)

Calculates the average maximal integration cost per clause (according to the DLT with cancelled modifier weight configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Lexical Sophistication Feature: Kucera and Francis (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Kucera and Francis norm list of frequencies (1967), which is included in the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Pavio (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Pavio norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: 3d Person Verb

Calculates 3d person (singular and plural) verbs density of the text. 3d person verbs include for Italian: Vip3 Vii3 Vis3 Vif3 Vcp3 Vci3 Vdp3 VAip3 VAii3 VAis3 VAif3 VAcp3 VAci3 VAdp3 VMip3 VMii3 VMis3 VMif3 VMcp3 VMci3 VMdp3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
3dPersonVerbDensity = num3dPersonVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Appositional Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of appositional modifiers in the text. Gives an absolute number.

An appositional modifier of an NP is an NP immediately to the right of the first NP that serves to define or modify that NP. It includes parenthesized examples, as well as defining abbreviations in one of these structures.

Ex: "Marcello Pagliacci, direttore del centro" appos(Pagliacci, direttore) "il presidente Scalfaro" appos(Scalfaro, presidente) "piccole e medie imprese (PMI)" appos(imprese, PMI)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency Per Million Words (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure per million words) of functional words (FW).

Formula:

```
for each functional word type:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the open subtitles corpus ( number of lines in the SUBTLEX-[lang].csv file ) / 1000 000 )

sum of LFs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency Per Million Words (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure per million words) of functional words (FW).

Formula:

```
for each functional word token:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the open subtitles corpus ( number of lines in the SUBTLEX-[lang].csv file ) / 1000 000 )

sum of LFs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Copula

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of copulas in the text. Gives an absolute number.

A copula is the relation between the complement of a copular verb and the copular verb.

Ex: "Giovanni è un uomo onesto" cop(uomo, è)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Conditional Present Verb

Calculates conditional present verbs density of the text. Conditional present verbs include for Italian: Vdp Vdp3 VAdp VAdp3 VMdp VMdp3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
conditionalPresentVerbDensity = numConditionalPresentVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Minimal Age of Active Use in KCT (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Minimal Age of Active Use in KCT list of lexical words (LW).

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Determiner DT

Calculates determiner DT density of the text. Determiner DTs include the Penn Treebank tag DT.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
DTDensity = numDT / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Negation Adverb

Calculates negation adverb density of the text. Negation adverb for Italian: BN

Availability:

```
This feature is NOT available for English, German.
```

Formula:

```
negationAdvDensity = numNegationAdverbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Complex Prepositional Phrases per Clause

Calculates the syntactic complexity of the text. Calculates the complex prepositional phrases per clause.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of complex prepositional phrases / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CPP_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency Per Million Words (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure per million words) of all words (AW).

Formula:

```
for each word type:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the
open subtitles corpus ( number of lines in the SUBTLEX-[lang].csv file ) / 1000
000 )

sum of LFs/ number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Purpose Connectives

Calculates the number of purpose connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The lists contains 6 connectives: acciocché affinché al fine di allo scopo che allo scopo di onde

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

POS Density Feature: Non Finite Verb

Calculates Non Finite Verb density of the text. Non Finite Verbs include for German: the Tiger tags VVINF VVIZU VVPP VAINF VAPP VMINF VMPP. for Italian: the tags Vg Vp Vf Vm VAg VAp VAf VAm VMg VMp VMf VMm.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
nonFinVerbDensity = numNonFinVerb / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (original cost configuration)

Calculates the average frequency of high adjacent IC (according to the DLT with its original cost configuration) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Number of Connectives: Additive Connectives (Breindl for German)

Calculates the number of additive connectives according to Breindl for German, to Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The German list contains 9 connectives.

Example: außerdem ebenfalls ebenso überdies übrigens und

The Italian list contains 17 connectives.

Example: anche ancora e ed in aggiunta in oltre in più

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Cohesive Complexity Feature: Adversative Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the adversative connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The German list contains 10 connective groups.

Example: zwar : zwar-aber,zwar-doch aber sondern : sondern-auch nur : nicht-nur einerseits

The Italian lists contains 38 connectives.

Example: al contrario all'inverso all'opposto anzi anziché ciononostante

Formula:

```
number of adversative connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

POS Density Feature: Adjective

Calculates adjective density of the text. Adjectives include for English: the Penn Treebank tags JJ, JJR and JJS, for German: the Tiger tags ADJA and ADJD, for Italian: As Ap An APs APp APn.

Formula:

```
adjDensity = numAdjectives / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Past Participle Verb

Calculates past participle verb density of the text. Past particple verbs include for English: the Penn Treebank tag VBN, for German: the Tiger tags VVPP VMPP VAPP, for Italian: the tags Vp VAp VMp

Formula:

```
VBNDensity = numVBNs /  numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Verb

Calculates the verb density of the text. Verbs include for English: the Penn Treebank tags VB, VBD, VBG, VBN, VBP, VBZ, for German: the Tiger tags VVFIN VVIMP VVINF VVIZU VMFIN VMIMP VMINF VMIZU VMPP VAFIN VAIMP VAINF VAIZU VAPP for Italian: Vip, Vip3, Vii, Vii3, Vis, Vis3, Vif, Vif3, Vcp, Vcp3, Vci, Vci3, Vdp, Vdp3, Vg, Vp, Vf, Vm, VMip, VMip3, VMii, VMii3, VMis, VMis3, VMif, VMif3, VMcp, VMcp3, VMci, VMci3, VMdp, VMdp3, VMg, VMp, VMf, VMm

Formula:

```
verbDensity = numVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Explicative Connectives per Token

Calculates the cohesive complexity of the text. Calculates the explicative connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian. The list contains 30 connectives.

Example: ad esempio appunto cioè come concretamente davvero

Formula:

```
number of explicative connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

POS Density Feature: Indicative Past Tense Verb

Calculates past tense verb density of the text. Past tense verbs include for Italian: indicative past (imperfect excluded): Vis Vis3 VAis Vis3 VMis VMis3.

Availability:

```
This feature is NOT available for: German, English
```

Formula:

```
VBDDensity = numVBDs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Clausal Complement

Counts the number of clausal complements in the text. Gives an absolute number.

A clausal complement of a verb or adjective is a dependent clause with an internal subject which functions like an object of the verb, or adjective. Clausal complements for nouns are limited to complement clauses with a subset of nouns like "fact" or "report". We analyze them the same (parallel to the analysis of this class as "content clauses" in Huddleston and Pullum 2002). Such clausal complements are usually finite (though there are occasional remnant English subjunctives)

Ex: "chiedere all'uomo che modifichi il suo comportamento" ccomp(chiedere, modifichi) "nessuno ha mai capito perché si è divertito a sabotare l'incontro" ccomp(capito, divertito)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Connectives: Causal Connectives (Breindl for German)

Calculates the number of causal connectives listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The Italian list contains 16 connectives.

Example: ché considerato che dacché dal momento che dappoiché dato che

The German list contains 8 connectives: denn : geschweige denn also deshalb deswegen folglich nämlich somit weil

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Syntactic Complexity Feature: Auxiliary Passives per Sentence

Calculates the syntactic complexity of the text. Calculates the number of auxiliary passives per sentence.

A passive auxiliary of a clause is a non-main verb of the clause which contains the passive information

Ex: "Significa che sono state tagliate le vie nervose" auxpass(tagliate, state) "Le donne candidate vengono ritenute intelligenti" auxpass(ritenute, vengono)

Formula:

```
number of auxiliary passives / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AuxiliaryPassive_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Number of Syntactic Constituents: Discourse Element

Counts the number of discourse elements in the text. Gives an absolute number.

This is used for interjections and other discourse particles and elements (which are not clearly linked to the structure of the sentence, except in an expressive way). We generally follow the guidelines of what the Penn Treebanks count as an INTJ. They define this to include: interjections (oh, uh-huh, Welcome), fillers (um, ah), and discourse markers (well, like, actually, but not you know)

Ex: Carlo è in Argentina :-) discourse(è, :-)) : Tint ha dato 'name(Argentina-4, :-)-5)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Verb Indicative Future

Calculates indicative future verbs density of the text. Indicative future verbs include for Italian: Vif Vif3 VAif VAif3 VMif VMif3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
indicativeFutureVerbDensity = numIndicativeFutureVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Concessive Connectives per Token (Eisenberg for German)

Calculates the cohesive complexity of the text. Calculates the concessive connectives per token, listed by Eisenberg for German. The list contains 38 connectives.

Example: trotz : trotz allem unbeschadet abgesehen von ungeachtet obwohl obgleich

Formula:

```
number of concessive connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Concessive_Feature.xml
POSDensity_NonPunctuationWords.xml
```

POS Density Feature: Twitter Tag

Calculates twitter tags density of the text. Twitter tags include for English the Penn Treebank tag SYM, for Italian: XM (twitter mentions ex.: @obama) XH (twitter hashtags ex.: #nlp).

Availability:

```
This feature is NOT available for: German
```

Formula:

```
twitterTagsDensity = numTwitterTags / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Negation Modifier

Counts the number of negation modifiers in the text. Gives an absolute number.

The negation modifier is the relation between a negation word and the word it modifies

Ex: "Non mi piace" neg(piace, non)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Richness: Type Token Ratio (TTR)

Calculates the type token ratio of a text. A word type is a non-duplicated token.

Formula:

```
This features calculates the TTR with the ### Formula:
TTR = T/N

T stands for number of word types, N for number of tokens.
```

Bib. ref.: Templin, M. (1957). Certain language skills in children. Minneapolis: University of Minnesota Press.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

Lexical Sophistication Feature: Meaningfulness Colerado (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Colerado norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Imageability (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Imageability norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word types found in the norm list / number of
functional word types of the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Conjunctive Imperfect Verb

Calculates conjunctive imperfect verbs density of the text. Conjunctive imperfect verbs for Italian: Vci Vci3 VAci VAci3 VMci VMci3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
conjunctiveImperfectVerbDensity = numConjunctiveImperfectVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Coordinate Phrases per Clause

Calculates the syntactic complexity of the text. Calculates the coordinate phrases per clause.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of coordinate phrases / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CP_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

POS Density Feature: Superlative Adverb

Calculates superlative adverb density of the text. Superlative adverbs include for English the Penn Treebank tag RBR.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
RBSDensity = numRBSs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Frequency (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list of all words (AW).

Formula:

```
sum of FREQCOUNTs of word tokens / number of word tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Flesch-Kincaid Grade Level

Calculates the Flesch-Kincaid Grade Level measure of readability.

Formula:

```
( 0.39 * mean_sentence_length_in_tokens ) + ( 11.8 *
mean_token_length_in_syllables ) - 15.59
```

This feature was originally defined and optimised for English. CTAP allows to apply it also to German and Italian, but it's up to the user to decide to what extent the results are reliable.

Bib. ref.: Kincaid JP, Fishburne RP Jr, Rogers RL, Chissom BS (February 1975). "Derivation of new readability formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy enlisted personnel" (PDF). Research Branch Report 8-75, Millington, TN: Naval Technical Training, U. S. Naval Air Station, Memphis, TN.

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
```

Number of Syntactic Constituents: Fragment Clauses

Counts the number of fragment clauses in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Mean Sentence Length in Tokens

Calculates the mean sentence length in number of letters.

Formula:

```
sum of the lengths of all the sentences of the text / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
LetterAnnotator.xml
NLetterFeature.xml
```

Lexical Sophistication Feature: SUBTLEX Contextual Diversity (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Contextual Diversity measure) of functional words (FW).

Formula:

```
for each functional word token:
        CD = number of films in which the word appears

sum of CDs / number of functional word tokens
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Age of Acquisition norm list by Gilhooly and Logie (1980), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of AoA values for lexical words found in the norm list / number of lexical
words of the text found in the norm list

Lexical words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

SD Sentence Length in Letters

Calculates the standard deviation of sentence length in number of letters.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
LetterAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Contextual Diversity (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Logarithmic Contextual Diversity measure) of all words (AW).

Formula:

```
for each token:
        CD = log10 (number of films in which the word appears + 1)

sum of CDs / number of tokens
```

Availability:

```
This feature is NOT available for German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency Per Million Words (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure per million words) of lexical words (LW).

Formula:

```
for each lexical word type:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the
Google corpus ( number of lines in the file ) / 1000 000 )

sum of LFs / number of lexical word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Frequency (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words

Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list of functional words (FW).

Formula:

```
sum of FREQCOUNTs of functional word types / number of functional word types
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Parataxis

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of parataxis relations in the text. Gives an absolute number.

The parataxis relation (from Greek for \place side by side") is a relation between the main verb of a clause and other sentential elements, such as a sentential parenthetical, a clause after a ":" or a ";", or two sentences placed side by side without any explicit coordination or subordination.

Ex: "Eppure, prosegue la lettera, a tutti è noto il fatto" parataxis(è, prosegue) Tint ha dato: 'parataxis(noto-10, prosegue-3)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Preconjunct

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of preconjuncts in the text. Gives an absolute number.

A preconjunct is the relation between the head of an NP and a word that appears at the beginning bracketing a conjunction (and puts emphasis on it), such as "either", "both", "neither").

Ex: "sia per le imprese editoriali, sia per le agenzie di stampa, i costi sono alti" preconj(imprese, sia) Tint non ha dato niente per la coppia (imprese, sia)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Brown (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Brown norm list of frequencies by Brown (1984), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Pavio (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Pavio norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Auxiliary Verb

Calculates Auxiliary Verb density of the text. Auxiliary Verbs include for German: the Tiger tags VAFIN VAIMP VAINF VAPP. for Italian: VAip,VA,VAip3, VAii, VAii3, VAis, Vis3, VAif, VAif3, VAcp, VAcp3, VAci, VAci3, VAdp, VAdp3, VAg, VAp, VAf, VAm

Availability:

```
This feature is NOT available for English.
```

Formula:

```
auxVerbDensity = numAuxVerb / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Connectives: Multifunctional Connectives

Calculates the number of multifunctional connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The lists contains 23 connectives.

Example: The connectives 'dopo' and 'poi' both have 2 meanings, temporal and hierarchical. So they appear in 3 lists: temporal, hierarchical and multifunctional.

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

POS Density Feature: Plural Adjective

Calculates adjective density of the text. Plural adjectives include for Italian: Ap APp

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
adjDensity = numPluralAdjectives / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Adversative and Concessive Connectives per Token (Eisenberg for German)

Calculates the cohesive complexity of the text. Calculates the adversative and concessive connectives per token, listed by Eisenberg for German. The lists contains 78 connectives.

Example: gegen entgegen zuwider statt : statt dass

Formula:

```
number of adversative and concessive connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

POS Density Feature: Relative Pronoun

Calculates Relative Pronoun density of the text. Relative Pronouns include for German: the Tiger tag PRELS PRELAT, for Italian: PR.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
relPronDensity = numRelPron / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Fragment T-units

Counts the number of fragment T-units in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Richness: Type Token Ratio (Log TTR)

Calculates the type token ratio of a text. A word type is a non-duplicated token.

Formula:

```
This features calculates the Bilogarithmic TTR with the ### Formula:
LogTTR = LogT/LogN

T stands for number of word types,
N stands for number of tokens.
```

Bib. ref.: Herdan, G. 1960. Type-Token Mathematics: A Textbook of Mathematical Linguistics. The Hague: Mouton.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

Lexical Sophistication Feature: Concreteness (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Concreteness norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical words found in the norm list / number of lexical words
of the text found in the norm list
```

```
Lexical words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Concreteness (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Concreteness norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Dependent clause ratio

Calculates the syntactic complexity of the text. Calculates the dependent clause ratio.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of dependent clauses / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_DC_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

POS Density Feature: Indefinite Pronoun

Calculates Indefinite Pronoun density of the text. Indefinite Pronouns include for German: the Tiger tags PIAT PIDAT. for Italian: the tag PI.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
indefPronDensity = numIndefPron / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure) of functional words (FW).

Formula:

```
for each functional word token:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: All Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the number of connectives per Token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The list for German contains 69 connectives. The list for Italian contains 212 connectives.

Formula:

```
number of connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSDensity_NonPunctuationWords.xml
NConnectives_Breindl_All_Feature.xml
```

POS Density Feature: Modal Verb

Calculates Modal Verb density of the text. Modal Verbs include for German: the Tiger tags VMFIN VMINF VMPP. for Italian: VMip VMip3 VMii VMii3 VMis VMis3 VMif VMif3 VMcp VMcp3 VMci VMci3 VMdp VMdp3 VMg VMp VMf VMm

Availability:

```
This feature is NOT available for: English
```

Formula:

```
modalVerbDensity = numModalVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Contextual Diversity (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Contextual Diversity measure) of all words (AW).

Formula:

```
for each token:
        CD = number of films in which the word appears
sum of CDs / number of tokens
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Foreign Word

Calculates foreign word density of the text. Foreign words include for English: the Penn Treebank tag FW, for German: the Tiger tag FM, for Italian: the tags SWs, SWp, SWn.

Formula:

```
FWDensity = numFWs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Informativeness Per Million Words (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Informativeness Per Million Words) of lexical words (LW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each word type:
        WI = number of words with the same first 3 characters and of the same
length as this word (this word included)

sum of WIs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Familiarity Per Million Words (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Familiarity Per Million Words) of lexical words (LW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each lexical word type:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word (this word’s FREQCOUNT added to the sum)

sum of FAs / number of lexical word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Word Types with 2 or more Syllables

Calculates number of words types with 2 or more syllables. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (All Lemmas)

Calculates lexical sophistication of the text.

A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

All lemmas features take into consideration all lemmas, while unique lemmas features calculate only unique lemmas.

This feature calculates lexical sophistication with the Age of Acquisition norm list from the Burani et al., 2001, containing 626 Italian nouns (https://www.istc.cnr.it/en/grouppage/varless).

Formula:

```
sum of AoA values for lemmas of the text found in Burani's list / number of
lemmas of the text found in Burani's list

Words of the text that are not in Burani's list are ignored.
```

Bib. ref.: Cristina Burani, Lisa S Arduino, Laura Barca, "Una base di dati sui valori di età di acquisizione, frequenza, familiarità, immaginabilità, concretezza, e altre variabili lessicali e sublessicali per 626 nomi dell'italiano", Giornale Italiano di Psicologia, January 2001

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
```

Lexical Richness: Type Token Ratio (Uber)

Calculates the type token ratio of a text. A word type is a non-duplicated token.

Formula:

```
This features calculates the Uber index with the ### Formula:
TTR = (LogN)^2/Log(N/T)

T stands for number of word types,
N stands for number of tokens.
```

Bib. ref.: Dugast, D. (1979). Vocabulaire et stylistique. I Théâtre et dialogue [Vocabulary and style. Vol. 1 Theatre and dialogue]. Slatkine-Champion, Geneva, Switzerland.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure) of functional words (FW).

Formula:

```
for each functional word type:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (no modifier weight + less coordination weight + added verb weight)

Calculates the average frequency of high adjacent IC (according to the DLT with cancelled modifier weight, reduced coordination weight, and increased verb weight) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Number of Connectives: Other Connectives (Eisenberg for German)

Calculates the number of unspecified connectives connectives listed by Eisenberg. Gives an absolute number.

The list contains 10 connectives: jedoch sowie falls hierdurch höchstens oder ohnehin sofern sonst sprich

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Dependency Locality Theory: Total IC at Finite Verb (no modifier weight + less coordination weight + added verb weight)

Calculates the total integration cost (according to the DLT with cancelled modifier weight, reduced coordination weight, and increased verb weight) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Cohesive Complexity Feature: Other Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. This feature calculates the other connectives listed by Breindl per token: connectives that are not included in lists with defined categories.

The lists contains 10 connectives: jedoch sowie falls hierdurch höchstens oder ohnehin sofern sonst sprich

Formula:

```
number of other connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Other_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (less coordination weight)

Calculates the average frequency of high adjacent IC (according to the DLT with reduced coordination weight configuration) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Syntactic Complexity Feature: T-unit complexity ratio

Calculates the syntactic complexity of the text. Calculates the T-unit complexity ratio.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of clauses / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_C_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Number of Syntactic Constituents: Adjectival Clause Modifier

Counts the number of adjectival clause modifiers in the text. Gives an absolute number.

Ex: Negozi aperti Evacuata la Tate Gallery. qualcosa da bere

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Verb Cluster per Clause

Calculates the syntactic complexity of the text. Calculates the verb cluster per clause.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of verb cluster / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_VC_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Lexical Sophistication Feature: Google Books Word Familiarity Per Million Words (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Familiarity Per Million Words) of all words (AW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each word type:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word (this word’s FREQCOUNT added to the sum).

sum of FAs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Contextual Diversity (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Contextual Diversity measure) of lexical words (LW).

Formula:

```
for each lexical word token:
        CD = number of films in which the word appears

sum of CDs / number of lexical word tokens
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Additive Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the additive connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The German list contains 9 connectives.

Example: außerdem ebenfalls ebenso überdies übrigens und

The Italian list contains 17 connectives.

Example: anche ancora e ed in aggiunta in oltre in più

Formula:

```
number of additive connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
NConnectives_Breindl_Additive_Feature.xml
```

Lexical Sophistication Feature: SUBTLEX Contextual Diversity (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Contextual Diversity measure) of functional words (FW).

Formula:

```
for each functional word type:
        CD = number of films in which the word appears

sum of CDs / number of functional word types
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Age of Acquisition norm list by Gilhooly and Logie (1980), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of AoA values for word types found in the norm list / number of word types
of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Mean Age of Active Use in KCT (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Mean Age of Active Use in KCT list of all words (AW).

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Indicative Verb

Calculates indicative verbs density of the text. Indicative verbs include for Italian: Vip Vip3 Vii Vii3 Vis Vis3 Vif Vif3 VAip VAip3 VAii VAii3 VAis VAis3 VAif VAif3 VMip VMip3 VMii VMii3 VMis VMis3 VMif VMif3

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
indicativeVerbDensity = numIndicativeVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Single-Word Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the single-word connectives listed by Breindl per token.

The lists contains 60 single-word connectives.

Example: aber allerdings also anfangs außerdem bald

Formula:

```
number of single-word connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
NConnectives_Breindl_AllSingle_Feature.xml
```

POS Density Feature: Superlative Adjective

Calculates superlative adjective density of the text. Superlative adjectives include for English the Penn Treebank tag JJS.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
JJSDensity = numJJS / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Thorndike Lorge (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Thorndike-Lorge norm list of frequencies (1944), which is included in the MRC Psycholinguistic Database (http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html).

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Informativeness Per Million Words (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Informativeness Per Million Words) of lexical words (LW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each word type:
        WI = number of words with the same first 3 characters and of the same
length as this word in the subtlex-[LANG].csv file (this word included)

sum of WIs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Pronoun

Calculates pronoun density of the text. Pronouns include for English: the Penn Treebank tags PRP, PRP$, WP, WP$, for German the Tiger tags: PDS PDAT PIS PIAT PIDAT PPER PPOSS PPOSAT PRELS PRELAT PRF PWS PWSAT PWAV PAV, for Italian: PC PD PE PI PP PQ PR.

Formula:

```
pronounDensity = numPronouns / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Dependency Locality Theory: Total IC at Finite Verb (less coordination weight)

Calculates the total integration cost (according to the DLT with reduced coordination weight configuration) at the finite verb.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Number of Syntactic Constituents: Prenominal Noun Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of prenominal noun modifiers in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Relative Clauses per Clause

Calculates the syntactic complexity of the text. Calculates the relative clauses per clause.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of relative clauses / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_RS_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure) of all words (AW).

Formula:

```
for each word type:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Wh-Determiner

Calculates wh-determiner density of the text. Wh-determiners include for English: the Penn Treebank tag WDT.

Availability:

```
This feature is NOT available for: German, Italian
```

Formula:

```
WDTDensity = numWDT / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Punctuation

Calculates Punctuation density of the text. Punctuation includes for German: the Tiger tags $, $. $(for Italian: FB FC FF FS

Availability:

```
This feature is NOT available for English.
```

Formula:

```
punctDensity = numPunct / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Dependent clauses per T-unit

Calculates the syntactic complexity of the text. Calculates the dependent clauses per T-unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of dependent clauses / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_DC_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Lexical Sophistication Feature: Kucera and Francis (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Kucera and Francis norm list of frequencies (1967), which is included in the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word types found in the norm list / number of
functional word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Brown (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Brown norm list of frequencies by Brown (1984), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical words found in the norm list / number of lexical words
of the text found in the norm list
```

```
Lexical words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Relative Clause Modifier

Counts the number of relative clause modifiers in the text. Gives an absolute number.

Ex: "I saw the man you love" rcmod(man, love); "I saw the book which you bought" rcmod(book,bought) "distinzioni che possono rendere difficile un confronto" rcmod(distinzioni, rendere)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Mean Sentence Length in Syllables

Calculates the mean sentence length in syllables.

Formula:

```
sum of the lengths of all the sentences of the text / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
SyllableAnnotator.xml
NSyllableFeature.xml
LetterAnnotator.xml
NLetterFeature.xml
```

Number of Tokens with 2 or more Syllables

Calculates number of words tokens with 2 or more syllables. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
```

POS Density Feature: WH Adverb

Calculates WH adverb density of the text. WH adverbs include for English: the Penn Treebank tag WRB.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
WRBDensity = numWRBs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure) of all words (AW).

Formula:

```
for each word type:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of word types
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Familiarity Per Million Words (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Familiarity Per Million Words) of functional words (FW).

Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each functional word type:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word (this word’s FREQCOUNT added to the sum)

sum of FAs / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Contextual Diversity (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Logarithmic Contextual Diversity measure) of lexical words (LW).

Formula:

```
for each lexical word token:
        CD = log10 (number of films in which the word appears + 1)

sum of CDs / number of lexical words
```

Availability:

```
This feature is NOT available for German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Auxiliaries per Sentence

Calculates the syntactic complexity of the text. Calculates the number of auxiliaries per sentence.

An auxiliary of a clause is a non-main verb of the clause, e.g., a modal auxiliary, or a form of "be", "do" or "have" in a periphrastic tense

Ex: "quella che ha fatto" aux(fatto, ha) "era convinto di dover vivere" aux(vivere, dover)

Formula:

```
number of auxiliaries / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_Auxiliary_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Google Books Word Frequency (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list of functional words (FW).

Formula:

```
sum of FREQCOUNTs of functional word tokens / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Coordinating Conjunction

Calculates conordinating conjunction density of the text. Coordinating conjunction include for English: the Penn Treebank tags CC, for German: the Tiger tag KON, for Italian: CC tag.

Formula:

```
CCDensity = numCC /     numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Mean Age of Active Use in KCT (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Mean Age of Active Use in KCT list of all words (AW).

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Argumentative Connectives

Calculates the number of argumentative connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The lists contains 5 connectives: a proposito di in relazione a per quanto riguarda relativamente a riguardo a

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

POS Density Feature: Particle

Calculates particle density of the text. Particles include for English: the Penn Treebank tag RP, for German: the Tiger tags PTKZU PTKNEG PTKVZ PTKANT PTKA

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
RPDensity = numRPs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Complex Prepositional Phrases per Sentence

Calculates the syntactic complexity of the text. Calculates the complex prepositional phrases per sentence.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of complex prepositional phrases / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CPP_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency Per Million Words (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure per million words) of lexical words (LW).

Formula:

```
for each lexical word type:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the
open subtitles corpus ( number of lines in the SUBTLEX-[lang].csv file ) / 1000
000 )

sum of LFs / number of lexical word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Familiarity Per Million Words (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Familiarity Per Million Words) of lexical words (LW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each lexical word token:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word in the subtlex-[LANG].csv file (this word’s
FREQCOUNT added to the sum)

sum of FAs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Imperative Verb

Calculates imperative verbs density of the text. Imperative verbs include for Italian: Vm VAm VMm

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
imperativeVerbDensity = numImperativeVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Mean Length of T-unit

Calculates the syntactic complexity of the text. Calculates the mean length of T-unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of words / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_T_Feature.xml
```

Lexical Variation Feature: Verb Variation 1

Calculates lexical variation of the text. Lexical words are certain types of verbs, nouns, adjectives, and adverb. This feature calculates Verb variation.

Formula:

```
VV1 = nVerbType / nVerbToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Thorndike Lorge (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Thorndike-Lorge norm list of frequencies (1944), which is included in the MRC Psycholinguistic Database (http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html).

Formula:

```
sum of values for functional word types found in the norm list / number of
functional word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Variation Feature: Corrected Verb Variation 1

Calculates lexical variation of the text. Lexical words are certain types of verbs, nouns, adjectives, and adverb. This feature calculates Verb variation.

Formula:

```
CVV1 = nVerbType / sqrt(2 * nVerbToken)
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Frequency (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list of lexical words (LW).

Formula:

```
sum of FREQCOUNTs of lexical word types / number of lexical word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Nominal Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of nominal modifiers in the text. Gives an absolute number.

Tint: L'allarme è scattato la scorsa settimana.

Dependency Parse (enhanced plus plus dependencies): root(ROOT-0, scattato-4) det(allarme-2, L'-1) nsubj(scattato-4, allarme-2) aux(scattato-4, è-3) det(settimana-7, la-5) amod(settimana-7, scorsa-6) nmod(scattato-4, settimana-7) punct(scattato-4, .-8)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Cohesive Complexity Feature: Multi-Word Connectives per Connective (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the multi-word connectives per connective listed by Breindl. The lists contains 9 connectives.

Example: bevor : bevor nicht denn : geschweige denn nur : nicht-nur

Formula:

```
number of multi-word connectives / number of connectives
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_AllMulti_Feature.xml
NConnectives_Breindl_All_Feature.xml
```

POS Density Feature: Preposition

Calculates preposition density of the text. Prepositions include for English: the Penn Treebank tags IN, for German: the Tiger tags APPR APPRART APPO APZR, for Italian: E EA.

Formula:

```
PrepositionDensity = numPrepositions / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Colerado (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes

a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Colerado norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of
the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Comparative Adverb

Calculates comparative adverb density of the text. Comparative adverbs include for English the Penn Treebank tag RBR.

Availability:

```
This feature is NOT available for: German, Italian.
```

Formula:

```
RBRDensity = numRBRs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Proper Noun

Calculates Proper Noun density of the text. Proper nouns include for German: the Tiger tag NE, for Italian: SP.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
NEDensity = numNE / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Word Familiarity Per Million Words (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Familiarity Per Million Words) of functional words (FW). Familiarity is the cumulative frequency of all types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each functional word token:
        FA = sum of FREQCOUNTs of all the words with the same first 3 characters
and of the same length as this word in the subtlex-[LANG].csv file (this word’s
FREQCOUNT added to the sum)

sum of FAs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Functional Words

Calculates functional word density of the text. Functional words include for English: the Penn Treebank tags CC, IN, PDT, DT, WDT, PRP, PRP$, WP, WP$, CD, EX, FW, LS, MD, POS, RP, SYM, TO, UH; for German: the Tiger tags CARD ITJ KOUI KOUS KON KOKOM PDS PDAT PIS PIAT PIDAT PPER PPOSS PPOSAT PRELS PRELSAT PRF PWS PWAT PWAV PAV PTKZU PTKNEG PTKVZ PTKANT PTKA VAFIN VAIMP VAINF VAIZU VAPP TRUNC; for Italian: CC CS DD DE DI DQ DR E EA RD RI T

Formula:

```
functionalWordDensity = numFunctionalWords / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Thorndike Lorge (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Thorndike-Lorge norm list of frequencies (1944), which is included in the MRC Psycholinguistic Database (http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html).

Formula:

```
sum of values for lexical word types found in the norm list / number of lexical
word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for German, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Pavio (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Pavio norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for tokens found in the norm list / number of tokens of the text
found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Age of Acquisition norm list by Gilhooly and Logie (1980), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of AoA values for functional words found in the norm list / number of
functional words of the text found in the norm list

Functional words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Mean Age of Active Use in KCT (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Mean Age of Active Use in KCT list of lexical words (LW).

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical
word tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Prepositional Phrase

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of prepositional phrases in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Imageability (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Imageability norm list from the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical
word tokens of the text found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Verb Cluster per T-Unit

Calculates the syntactic complexity of the text. Calculates the verb cluster per T-Unit.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of verb cluster / number of T-Units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_VC_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Lexical Sophistication: Percentage of Unique Lemmas Listed in the De Mauro Basic Dictionary (unique lemmas)

Calculates the percentage of unique lemmas listed in the De Mauro basic dictionary.

Formula:

```
( number of unique lemmas that are listed in the De Mauro dictionary / number of
unique lemmas (excluding punctuations) ) * 100
```

Bib. ref.: Il Nuovo vocabolario di base della lingua italiana. A cura di Tullio De Mauro. 23 dicembre 2016. https://dizionario.internazionale.it/nuovovocabolariodibase

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
LemmaAnnotator.xml
```

Lexical Sophistication Feature: Meaningfulness Colerado (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Meaningfulness Colerado norm list from the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional word types found in the norm list / number of
functional word types of the text found in the norm list

Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

SD Token Length in Syllables

Calculates the standard deviation of token length in number of syllables.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
SyllableAnnotator.xml
LetterAnnotator.xml
```

Syntactic Complexity Feature: Appositional Modifiers per Sentence

Calculates the syntactic complexity of the text. Calculates the number of Appositional modifiers per sentence.

An appositional modifier of an NP is an NP immediately to the right of the first NP that serves to define or modify that NP. It includes parenthesized examples, as well as defining abbreviations in one of these structures.

Ex: "Marcello Pagliacci, direttore del centro" appos(Pagliacci, direttore) "il presidente Scalfaro" appos(Scalfaro, presidente) "piccole e medie imprese (PMI)" appos(imprese, PMI)

Formula:

```
number of appositional modifiers / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AppositionalModifier_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: Google Books Logarithmic Word Frequency (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Log10 word frequency measure) of all words (AW).

Formula:

```
for each token:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Noun Phrase as Adverbial Modifier

Counts the number of noun phrases as adverbial modifiers in the text. Gives an absolute number.

This relation captures various places where something syntactically a noun phrase (NP) is used as an adverbial modifier in a sentence. These usages include: (i) a measure phrase, which is the relation between the head of an ADJP/ADVP/PP and the head of a measure phrase modifying the ADJP/ADVP; (ii) noun phrases giving an extent inside a VP which are not objects; (iii) financial constructions involving an adverbial or PP-like NP, notably the following construction $5 a share, where the second NP means \per share"; (iv) oating reflexives; and (v) certain other absolutive NP constructions. A temporal modifier (tmod) is a subclass of npadvmod which is distinguished as a separate relation.

Ex: "Costava sei lire" npadvmod(costava, lire); Tint ha dato: 'dobj(Costava-1, lire-3)' "a quota un milione" npadvmod(quota, milione); "tutte, una dopo l'altra" npadvmod(tutte, una)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure) of functional words (FW).

Formula:

```
for each functional word token:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of functional word tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Purpose Connectives per Token

Calculates the cohesive complexity of the text. Calculates the purpose connectives per token, listed by Nadezda Okinina and Lorenzo Zanasi for Italian. The lists contains 6 connectives: acciocché affinché al fine di allo scopo che allo scopo di onde

Formula:

```
number of purpose connectives / number of tokens
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

Lexical Sophistication Feature: Brown (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Brown norm list of frequencies by Brown (1984), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for functional words found in the norm list / number of functional
words of the text found in the norm list
```

```
Functional words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Variation Feature: Noun

Calculates lexical variation of the text. Lexical words are verbs, nouns, adjectives, and adverbs. This feature calculates Noun variation.

Formula:

```
nounVariation = nNounType / nLexicalToken
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Mean Age of Active Use in KCT (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Mean Age of Active Use in KCT list of functional words (FW).

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional word tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Mean Length of Clause

Calculates the syntactic complexity of the text. Calculates the mean length of clause.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of words / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_C_Feature.xml
```

Lexical Sophistication Feature: Kucera and Francis (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Kucera and Francis norm list of frequencies (1967), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical word tokens found in the norm list / number of lexical word tokens of the text found in the norm list
```

```
Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: Brown (AW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Brown norm list of frequencies by Brown (1984), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for word types found in the norm list / number of word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: All Connectives (Breindl for German)

Calculates the number of all connectives listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The list for German contains 69 connectives. The list for Italian contains 212 connectives.

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: Age of Acquisition (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Age of Acquisition norm list by Gilhooly and Logie (1980), which is included in the MRC Psycholinguistic Database. http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of AoA values for functional words found in the norm list / number of
functional words of the text found in the norm list
```

```
Functional words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Postnominal Noun Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of postnominal noun modifier in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Informativeness Per Million Words (AW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Informativeness Per Million Words) of all words (AW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each token:
        WI = number of words with the same first 3 characters and of the same
length as this word (this word included)

sum of WIs / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Sentences

Calculates the number of sentences in a text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
```

Lexical Sophistication Feature: Google Books Word Frequency (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list of functional words (FW).

Formula:

```
sum of FREQCOUNTs of functional word types / number of functional word types
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Complex Nominals per Clause

Calculates the syntactic complexity of the text. Calculates the complex nominals per clause.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of complex nominals / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CN_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Syntactic Complexity Feature: Prenominal Modifier per Complex Noun Phrase

Calculates the syntactic complexity of the text. Calculates the prenominal modifier per complex noun phrase.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
number of prenominal modifier / number of complex noun phrases
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_PreN_Feature.xml
NSyntacticConstituent_CN_Feature.xml
```

Number of Connectives: Breindl Multi-Word Connectives

Calculates the number of all multi-word connectives listed by Breindl. Gives an absolute number.

The lists contains 9 connectives.

Example: bevor : bevor nicht denn : geschweige denn nur : nicht-nur

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Syntactic Complexity Feature: Complex T-unit per Sentence

Calculates the syntactic complexity of the text. Calculates the ratio of complex t-units to sentences.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of complex T-units / number of sentences
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CT_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Number of Syntactic Constituents: Passive Nominal Subject

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of passive nominal subjects in the text. Gives an absolute number.

A passive nominal subject is a noun phrase which is the syntactic subject of a passive clause

Ex: "L'ex arbitro triestino è stato messo in castigo" nsubjpass(messo, arbitro)

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Syntactic Constituents: Prepositional Complement

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of prepositional complements in the text. Gives an absolute number.

This is used when the complement of a preposition is a clause or prepositional phrase (or occasionally, an adverbial phrase). The prepositional complement of a preposition is the head of a clause following the preposition, or the preposition head of the following PP

Ex: "I vigili del fuoco sono accorsi per domare il rogo" pcomp(per, domare); Tint ha dato: 'advcl:per(accorsi-6, domare-8)' "Sono rientrati per sempre in Italia" pcomp(per, sempre)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Syntactic Complexity Feature: Adjectival Clause Modifiers per Sentence

Calculates the syntactic complexity of the text. Calculates the number of Adjectival Clause Modifiers per sentence.

Ex: Negozi aperti Evacuata la Tate Gallery. qualcosa da bere

Formula:

```
number of ACLs / number of sentences
```

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_AdjectivalClauseModifier_Feature.xml
NSyntacticConstituent_S_Feature.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure) of functional words (FW).

Formula:

```
for each functional word type:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of functional word types
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Lemmas

Calculates the number of lemmas in the text. Gives an absolute number.

AAE dependency: SentenceAnnotator.xml TokenAnnotator.xml LemmaAnnotator.xml

Lexical Sophistication Feature: Google Books Word Informativeness Per Million Words (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Google Books (2000 for German and 2012 for Italian) word frequency list (Google Books Word Informativeness Per Million Words) of lexical words (LW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type.

Formula:

```
for each lexical word token:
        WI = number of words with the same first 3 characters and of the same
length as this word (this word included).

sum of WIs / number of lexical word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Explicative Connectives

Calculates the number of explicative connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The list contains 30 connectives.

Example: ad esempio appunto cioè come concretamente davvero

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Number of Connectives: Hypothetical and Conditional Connectives

Calculates the number of hypothetical and conditional connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The list contains 20 connectives.

Example: ammettendo che nel caso in cui partendo dal presupposto che purché qualora

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure) of lexical words (LW).

Formula:

```
for each lexical word token:
        LF = log10 ( FREQCOUNT + 1 )

sum of LFs / number of lexical word tokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

POS Density Feature: Comparative Conjunction

Calculates the density of comparative conjunctions in the text. Comparative conjunctions include for German: the Tiger tag KOKOM.

Availability:

```
This feature is NOT available for English, Italian.
```

Formula:

```
compConjDensity = numCompConj / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Coordinate Phrases

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of coordinate phrases in the text. Gives an absolute number.

Availability:

```
This feature is NOT available for Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Number of Connectives: Temporal Connectives (Breindl for German)

Calculates the number of temporal connectives listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The Italian lists contains 55 connectives.

Example: a questo punto alla fine allora allorché allorquando anni fa antecedente a

The German lists contains 26 connectives.

Example: dann : selbst dann wenn,auch dann wenn,sogar dann wenn,wenn-dann bevor : bevor nicht zuletzt : nicht zuletzt anfangs bald bereits

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Word Frequency Per Million Words (LW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Log10 word frequency measure per million words) of lexical words (LW).

Formula:

```
for each lexical word token:
        LF = log10 ( FREQCOUNT + 1 ) - log10 ( total number of words  in the
open subtitles corpus ( number of lines in the SUBTLEX-[lang].csv file ) / 1000
000 )

sum of LFs / number of lexical word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Verb Cluster

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of verb clusters in the text. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Singular Noun

Calculates singular noun density of the text. Singular nouns include for English: the Penn Treebank tag NN, for Italian: SS.

Availability:

```
This feature is NOT available for: German.
```

Formula:

```
NNDensity = numNN / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```

```
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Finite Verb

Calculates Finite Verb density of the text. Finite Verbs include for German: the Tiger tags VVFIN VVIMP VAFIN VAIMP VMFIN. for Italian: Vip,Vip3,Vii,Vii3,Vis,Vis3,Vif,Vif3,Vcp,Vcp3,Vci,Vci3,Vdp,Vdp3.

Availability:

```
This feature is NOT available for English.
```

Formula:

```
finiteVerbDensity = numFiniteVerbs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Syntactic Complexity Feature: Verb Phrases per Clause

Calculates the syntactic complexity of the text. Calculates the verb phrases per clause.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of verb phrases / number of clauses
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_VP_Feature.xml
NSyntacticConstituent_C_Feature.xml
```

Gulpease

Calculates the Gulpease measure of readability.

Formula:

```
89 + ( ((300 * number of sentences) - (10 * number of letters)) / number of
tokens )
```

The results vary from 0 to 100, where the value "100" indicates the highest readability and "0" indicates the lowest readability.

Usually text with Gulpease index less than 80 are difficult to read for those who have finished the elementary school less than 60 are difficult to read for those who have finished the middle school less than 40 are difficult to read for those who have finished the university

This feature was originally defined and optimised for Italian. CTAP allows to apply it also to German and English, but it's up to the user to decide to what extent the results are reliable.

Bib. ref.: Pietro Lucisano and Maria Emanuela Piemontese. 1988. Gulpease. Una formula per la predizione della difficoltà dei testi in lingua italiana. Scuola e Città, 3:57–68.

AAE dependency:

```
SentenceAnnotator.xml
NSentenceFeature.xml
TokenAnnotator.xml
NTokenFeature.xml
LetterAnnotator.xml
NLetterFeature.xml
```

Cohesive Complexity Feature: Multi- to Single-Word Connectives (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the ratio of multi- to single-word connectives listed by Breindl.

Formula:

```
number of multi-word connectives / number of single-word connectives
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_AllMulti_Feature.xml
NConnectives_Breindl_AllSingle_Feature.xml
```

Lexical Sophistication Feature: Brown (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each norm list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words A norm list is a list where a norm value that describes a certain aspect of a word is given to each word. Only words that are included in the norm list are calculated. Words that do not appear in the norm list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Brown norm list of frequencies by Brown (1984), which is included in the MRC Psycholinguistic Database.
http://websites.psychology.uwa.edu.au/school/MRCDatabase/mrc2.html

Formula:

```
sum of values for lexical words found in the norm list / number of lexical words
of the text found in the norm list
```

```
Lexical words that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Additive Connectives (Eisenberg for German)

Calculates the number of additive connectives for German according to Eisenberg. Gives an absolute number.

The list contains 41 connectives.

Example: einschließlich samt nebst inklusive zuzüglich ohne zu

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
```

```
TokenAnnotator.xml
```

Syntactic Complexity Feature: Coordinate Phrases per T-unit

Calculates the syntactic complexity of the text. Calculates the coordinate phrases per T-unit.

Availability:

```
This feature is NOT available for Italian.
```

Formula:

```
number of coordinate phrases / number of T-units
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
NSyntacticConstituent_CP_Feature.xml
NSyntacticConstituent_T_Feature.xml
```

Number of Syntactic Constituents: Quantifier Phrase Modifier

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of quantifier phrase modifiers in the text. Gives an absolute number.

A quantifier modifier is an element modifying the head of a QP constituent. (These are modifiers in complex numeric quantifiers, not other types of "quantification". Quantifiers like "all" become det.)

Ex: "Vi sono oltre 40 modelli messi in commercio" quantmod(40, oltre) Tint ha dato: 'advmod(40-4, oltre-3)'

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

POS Density Feature: Ordinal Number

Calculates cardinal number density of the text. Ordinal numbers include for Italian: NOs NOp NOn.

Availability:

```
This feature is NOT available for: English, German.
```

Formula:

```
ODDensity = numODs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Lexical Sophistication Feature: SUBTLEX Logarithmic Contextual Diversity (LW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (Logarithmic Contextual Diversity measure) of lexical words (LW).

Formula:

```
for each lexical word type:
        CD = log10 (number of films in which the word appears + 1)

sum of CDs / number of lexical word types
```

Availability:

```
This feature is NOT available for German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Lexical Richness: Type Token Ratio (Root TTR)

Calculates the type token ratio of a text. A word type is a non-duplicated token.

Formula:

```
This features calculates the root TTR with the ### Formula:
LogTTR = T/sqrt(N)

T stands for number of word types
N stands for number of tokens.
```

Bib. ref.: Guiraud, P. 1960. Problemes et Methodes de la Statistique Linguistique. Dordrecht: D. Reidel.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
NTokenFeature.xml
NTokenTypeFeature.xml
```

POS Density Feature: WH Pronoun

Calculates WH pronoun density of the text. WH pronouns include for English: the Penn Treebank tag WP.

Availability:

```
This feature is NOT available for: German, Italian
```

Formula:

```
WPDensity = numWPs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Adversative and Concessive Connectives per Token (Breindl for German)

Calculates the cohesive complexity of the text. Calculates the adversative and concessive connectives per token, listed by Breindl for German, by Nadezda Okinina and Lorenzo Zanasi for Italian.

The German list contains 17 connectives.

Example: zwar : zwar-aber,zwar-doch aber sondern : sondern-auch nur : nicht-nur

The Italian lists contains 38 connectives.

Example: al contrario all'inverso all'opposto anzi anziché cionondimeno

Formula:

```
number of adversative and concessive connectives / number of tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_Temporal_Feature.xml
POSDensity_NonPunctuationWords.xml
```

POS Density Feature: List Item Marker

Calculates list item marker density of the text. List item markers include for English the Penn Treebank tag LS.

Availability:

```
This feature is NOT available for: German, Italian
```

Formula:

```
LSDensity = numLSs / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

POS Density Feature: Common Noun

Calculates Common Noun density of the text. Common Nouns include for German: the Tiger tag NN.

Availability:

```
This feature is NOT available for English and Italian.
```

Formula:

```
NNDensity = numNN / numTokens
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NTokenFeature.xml
POSAnnotator.xml
```

Number of Syntactic Constituents: Phrasal Verb Particle

Calculates the number of a specific syntactic constituents in the text. This feature counts the number of phrasal verb particles in the text. Gives an absolute number.

Used for clitic pronoun in reflexive verbs.

Ex: "appostar-si nel parcheggio" prt(appostar-, si) "si divertì" prt(divertì, si)

Tint ha dato: Il principe si divertì.

Dependency Parse (enhanced plus plus dependencies): root(ROOT-0, divertì-4) det(principe-2, Il-1) nsubj(divertì-4, principe-2) expl(divertì-4, si-3) punct(divertì-4, .-5)

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
ParseTreeAnnotator.xml
```

Dependency Locality Theory: High Adjacent IC at Finite Verb (no modifier weight + less coordination weight)

Calculates the average frequency of high adjacent IC (according to the DLT with cancelled modifier weight and reduced coordination costs configuration) at the finite verb. High costs are defined as costs higher than 2 after integration.

Availability:

```
This feature is NOT available for English, Italian.
```

Bib.ref.:

```
Gibson, Edward. 2000. The dependency locality theory: A distance-based theory of
linguistic complexity. In: Image, language, brain, pp. 95–126.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
LemmaAnnotator.xml
MorphologicalTagAnnotator.xml
POSAnnotator.xml
DependencyParseAnnotator.xml
```

Lexical Sophistication Feature: Mean Age of Active Use in KCT (FW Type)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Mean Age of Active Use in KCT list of functional words (FW).

Formula:

```
sum of values for functional word types found in the norm list / number of
functional word types of the text found in the norm list
```

```
Word types that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Word Types

Calculates the number of word types. A word type is a non-duplicated token. Gives an absolute number.

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
TokenTypeAnnotator.xml
```

Lexical Sophistication Feature: Minimal Age of Active Use in KCT (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the Minimal Age of Active Use in KCT list of functional words (FW).

Formula:

```
sum of values for functional word tokens found in the norm list / number of
functional word tokens of the text found in the norm list

Tokens that are not in the norm list are ignored.
```

Availability:

```
This feature is NOT available for Italian, English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Cohesive Complexity Feature: Breindl Single-Word Connectives per Connective

Calculates the cohesive complexity of the text. Calculates the single-word connectives per connective listed by Breindl.

The lists contains 60 single-word connectives.

Example: aber allerdings also anfangs außerdem bald

Formula:

```
number of single-word connectives / number of connectives
```

Availability:

```
This feature is NOT available for English, Italian.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
NConnectives_Breindl_AllSingle_Feature.xml
NConnectives_Breindl_All_Feature.xml
```

Lexical Sophistication Feature: SUBTLEX Word Informativeness Per Million Words (FW Token)

Calculates lexical sophistication of the text. Three sophistication measures are calculated from each frequency list: AW: all words LW: lexical words, which are verbs, nouns, adverbs and adjectives FW: function words, which are non-lexical words Only words that are included in the frequency list are calculated. Words that do not appear in the frequency list are omitted.

Token features take into consideration all word tokens, while type features calculate only unique tokens.

This feature calculates lexical sophistication with the SUBTLEX word frequency list (SUBTLEX Word Informativeness Per Million Words) of functional words (FW). Informativeness is the number of types with the same initial character trigram and of the same length as the given type:

Formula:

```
for each functional word token:
        WI = number of words with the same first 3 characters and of the same
length as this word in the subtlex-[LANG].csv file (this word included)

sum of WIs / number of functional word tokens
```

Availability:

```
This feature is NOT available for English.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
POSAnnotator.xml
```

Number of Connectives: Consequence Connectives

Calculates the number of consequence connectives listed by Nadezda Okinina and Lorenzo Zanasi for Italian. Gives an absolute number.

The list contains 14 connectives.

Example: così che da ciò si deduce che di conseguenza dunque

Availability:

```
This feature is NOT available for English, German.
```

AAE dependency:

```
SentenceAnnotator.xml
TokenAnnotator.xml
```