1 Popis riešenia

Nasledujúci skript v jazyku Perl slúži na extrakciu kľúčových slov z textového súboru a ich zoradenie podľa dôležitosti.

```
use strict;
use warnings;

#!/usr/bin/perl

use strict;
use warnings;
use Lingua::Stem::Snowball;
use Getopt::Long;
use JSON;

# Parse command line options
my $stopwords_dir = "stopwords";
my $lang = "en";
my $num_keywords = 10;
my $output_format = "text";
my $help = 0;
GetOptions(
    'stopwords=s' => \$stopwords_dir,
    'lang=s' => \$lang,
    'num_keywords=i' => \$num_keywords,
    'output_format=s' => \$output_format,
    'help' => \$help,
) or die "Error parsing command line options\n";

# Check for help option
if ($help) {
    print "Usage: perl script.pl [options]\n";
    print "Extracts and ranks keywords from text
       provided on standard input.\n";
    print "Options:\n";
    print "  --stopwords DIR      Path to stop words
       folder (default: stopwords)\n";
    print "  --lang LANG          Language for
       stemming and stop words (default: en)\n";
    print "  --num_keywords NUM    Number of keywords
       to output (default: 10)\n";
    print "  --output_format FMT   Output format (text
       , json, csv) (default: text)\n";
    print "  --help                Show this help
       message\n";
```

```
36      exit;
37  }
38
39  # Load stop words from file
40  my $stopwords_file = "$stopwords_dir/$lang.txt";
41  my @stopwords;
42  if (-e $stopwords_file) {
43      open(my $fh, "<", $stopwords_file) or die "Cannot
            open file $stopwords_file: $!";
44      while (my $line = <$fh>) {
45          chomp $line;
46          push @stopwords, $line;
47      }
48      close($fh);
49  } else {
50      warn "Warning: Stop words file '$stopwords_file'
            not found. Proceeding without stop words.\n";
51  }
52
53  # Initialize stemmer
54  my %lang_map = (
55      'sk' => 'sk',
56      'cz' => 'cs',
57      'en' => 'en',
58  );
59  my $stemmer_lang = $lang_map{$lang} || 'en';
60  my $stemmer = Lingua::Stem::Snowball->new( lang =>
        $stemmer_lang );
61
62  # Load input text from standard input
63  my $text = do { local $/; <STDIN> };
64  if (!defined $text || length($text) == 0) {
65      die "Error: No input provided on standard input\n"
            ;
66  }
67
68  # Tokenize text into individual words
69  my @words = split /\s+/, $text;
70
71  # Remove punctuation and convert to lowercase
72  foreach my $word (@words) {
73      $word =~ s/[[:punct:]]//g;
74      $word = lc $word;
75  }
76
77  # Remove stop words and apply stemming
```

```
78 @words = grep { my $w = $_; !grep { $_ eq $w }
       @stopwords } @words;
79 foreach my $word (@words) {
80     $word = $stemmer->stem($word);
81 }
82
83 # Calculate word frequency
84 my %word_freq;
85 foreach my $word (@words) {
86     $word_freq{$word}++;
87 }
88
89 # Calculate word importance using TextRank algorithm
90 my %scores;
91 my $damping_factor = 0.85;
92 my $max_iterations = 50;
93 my $convergence_threshold = 0.0001;
94 my $num_words = scalar @words;
95
96 foreach my $word (keys %word_freq) {
97     $scores{$word} = 1;
98 }
99
100 for (my $i = 0; $i < $max_iterations; $i++) {
101     my %new_scores;
102
103     foreach my $word (keys %word_freq) {
104         $new_scores{$word} = (1 - $damping_factor) /
               $num_words;
105
106         foreach my $adj_word (keys %word_freq) {
107             next if $word eq $adj_word;
108
109             my $adj_frequency = $word_freq{$adj_word};
110             my $adj_links = $word_freq{$word};
111
112             if ($adj_frequency > 0) {
113                 $new_scores{$word} += ($damping_factor
                       * $scores{$adj_word} * $adj_links)
                       / $adj_frequency;
114             }
115         }
116     }
117
118     my $max_diff = 0;
119     foreach my $word (keys %scores) {
```

```
        my $diff = abs($scores{$word} - $new_scores{$word});
        if ($diff > $max_diff) {
            $max_diff = $diff;
        }
    }

    last if $max_diff < $convergence_threshold;

    %scores = %new_scores;
}

# Normalize scores
my $max_score = (sort { $b <=> $a } values %scores)[0];
foreach my $word (keys %scores) {
    $scores{$word} /= $max_score;
}

# Sort keywords by importance
my @sorted_words = sort { $scores{$b} <=> $scores{$a} } keys %scores;

# Output keywords to standard output
if ($output_format eq "json") {
    my @output_data;
    foreach my $word (@sorted_words[0..$num_keywords-1]) {
        if (defined $word) {
            push @output_data, { word => $word, score => $scores{$word} };
        }
    }
    print to_json(\@output_data, { pretty => 1 });
} elsif ($output_format eq "csv") {
    print "word,score\n";
    foreach my $word (@sorted_words[0..$num_keywords-1]) {
        if (defined $word) {
            print "$word,$scores{$word}\n";
        }
    }
} else {
    foreach my $word (@sorted_words[0..$num_keywords-1]) {
        if (defined $word) {
```

```
159                 print "$word\n";
160             }
161         }
162 }
```

1. Skript načíta textový súbor ako vstup zo štandardného vstupu.
2. Načítaný text sa tokenizuje na jednotlivé slová.
3. Pre každé slovo sa vykonajú úpravy, ako odstránenie interpunkcie a prevod na malé písmená.
4. Následne sa odstránia stop slová pre zvolený jazyk.
5. Pre každé slovo sa vypočíta frekvencia výskytu.
6. Dôležitosť slov sa vypočíta pomocou algoritmu TextRank.
7. Dôležitosť slov sa normalizuje a zoradia sa podľa hodnoty.
8. Nakoniec sa kľúčové slová vypíšu na štandardný výstup, každé slovo na samostatnom riadku.

Skript využíva moduly Lingua::Stem::Snowball pre stemming slov a Getopt::Long pre spracovanie príkazového riadku.

2 Inštalácia modulov

Pred použitím skriptu sa uistite, že máte nainštalované moduly Lingua::Stem::Snowball a Getopt::Long. Ak nie sú nainštalované, môžete ich nainštalovať pomocou správcu balíkov CPAN.

```
cpan Lingua::Stem::Snowball
cpan Getopt::Long
```

3 Použitie skriptu

Na spustenie skriptu použite nasledujúci príkaz:

```
perl klicova_slova.pl < input.txt
```

Kde `klicova_slova.pl` je názov skriptu a `input.txt` je vstupný textový súbor.