Condorcet Fusion: an implementation

Improved Retrieval through Rank Fusion

Marco Alfonso, Davide Martini, Giovanni Mazzocchin
February 9, 2018

Università
degli Studi
di Padova

Introduction

This project implements some ideas coming from a paper by **Aslam** and **Montague**.

These authors applied *Social Choice Theory* to **Rank Fusion**.

They claim that their algorithm beats the ones by **Fox** and **Shaw** most of the time.

Starter: basic fusion strategies

Fox and **Shaw** devised some simple, score-based fusion formulae:

CombMAX

fused score = *max(scores)*

CombMIN

fused score = *min(scores)*

CombSUM

fused score = *sum(scores)*

Starter: basic fusion strategies

Fox and **Shaw** devised some simple, score-based fusion formulae:

CombMED

$$fused\ score = median(scores)$$

CombANZ

$$fused\ score = \frac{CombSUM}{\#nonZeroScores}$$

CombMNZ

$$fused\ score = CombSUM \times \#nonZeroScores$$

Are we comparing apples and oranges?

Of course we can't perform any sensible fusion without **normalization**.
Lee came up with two handy formulae:

Max Norm

$$max\_norm = \frac{old\_score}{max\_score}$$

Min Max Norm

$$min\_max\_norm = \frac{old\_score - min\_score}{max\_score - min\_score}$$

What about voting systems?

Metaphor: *voters* are *retrieval systems*
candidates are *documents*

The algorithm inspired by the **Condorcet method** starts from a document comparator known as *Simple Majority Runoff*.

“Simple Majority Runoff”

Imagine a **contest** between two documents, then the rule below sounds sensible, doesn’t it?:

`The number of a document’s votes is directly related to the number of times it ranks above the other.`

A theoretical approach

The first proposed algorithm isn't well-suited for large collections. It's based on the computation of **Hamiltonian paths** on a graph and has a $O(n^2k)$ time complexity.