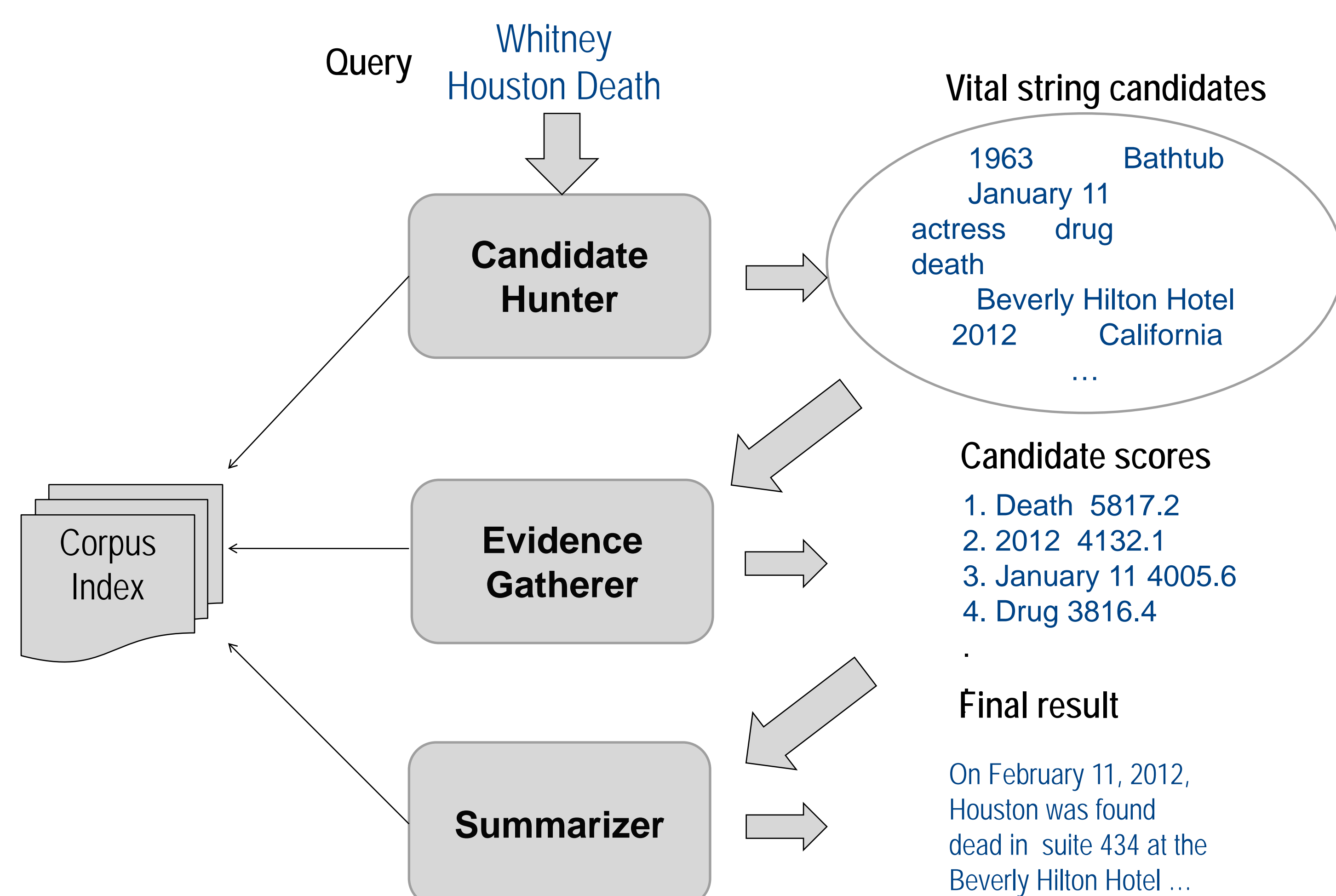


Hunter Gatherer: UDEM at 1CLICK-2

Pablo Duboue, Jing He, Jian-Yun Nie
DIRO, Université de Montréal, Canada

Introduction

- 1CLICK-2 Task^[1]: retrieve relevant *information Units* instead of relevant documents
- Hunter Gatherer System
 - ✓ DeepQA framework^[2] for vital string ranking
 - ✓ ILP^[3] or MMR^[4] for summarization



Candidate Hunter

- Input: Query String
- Output: Vital String Candidates
- Method
 - a) Query Parsing and Evaluation (main search)

- Detect named entities
- Build Indri query
- Evaluate query to get top ranked passages

e.g.
Query = "Whitney Houston death", "Whitney Houston" is a named entity
Indri Query = #combine[passage120:50](#1(Whitney Houston) death)

- b) Candidate Identification: identify candidates from top ranked passages
 - Tokens
 - Named entity
 - "key information" extractor learnt from Wikipedia infobox and article text with CRF model

References

- [1] M. P. Kato, M. Ekstrand-Abueg, V. Pavlu, T. Sakai, T. Yamamoto, and M. Iwata. Overview of the NTCIR-10 1CLICK-2 task. In NTCIR 2013.
- [2] D. C. Gondek, A. Lally, A. Kalyanpur, J. W. Murdock, P. A. Duboue, L. Zhang, Y. Pan, Z. M. Qiu, and C. Welty. A framework for merging and ranking of answers in DeepQA. IBM Journal of Research and Development, 56(3.4) 14:1-14:12, 2012.
- [3] J. Clarke and M. Lapata. Global inference for sentence compression: An integer linear programming approach. Journal of Artificial Intelligence Research, 31(1):399-429, 2008.
- [4] J. G. Carbonell and J. Goldstein. The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In SIGIR 1998.

Evidence Gatherer

- Input: Candidate Set
- Output: Candidate Scores
- Method:
 - a) Gather evidence
 - build Indri Query from the original query and candidate
 - Evaluate the query and get top ranked passages and scores

- a) Score candidates based on evidences

$$R(q, u) = \lambda_1 \cdot \sum_{p \in MS, u \in p} (R(q, p) + \alpha) + \lambda_2 \cdot \sum_{p \in ES} (R(q, p) + \beta)$$

Main search Evidence search

Summarizer: Organizing Candidates

- Input: Candidate Scores
- Output: Constraint length of text containing most relevant candidates
- Method:
 - MMR (Maximal Marginal Relevance) iterative algorithm to select sentences
 - score sum of candidates in a sentence as the relevant score
 - Jaccard distance of bigrams between sentences as redundancy score
 - ILP (Integer Linear Programming)
 - Objective function: score sum of candidates in the selected sentences
 - Constraint: the length of sentences is smaller than value k .

Evaluation

- Run 1. Hunter (candidate = tokens + NEs) + Gatherer + Summarizer (MMR)
- Run 3. Hunter (candidate = tokens + NEs + learnt extractor output) + Gatherer + Summarizer (MMR)
- Run 4. Hunter (candidate = tokens + NEs) + Gatherer + Summarizer (ILP)

Table. Evaluation Results for Desktop Mandatory Runs

Run / Category:	All	ACTOR	ATHLE	ARTIST	POLIT	FACIL	GEO	DEFIN	QA
Run 1	0.047	0.040	0.028	0.039	0.037	0.060	0.025	0.066	0.068
Run 3	0.050	0.058	0.016	0.038	0.086	0.058	0.016	0.077	0.053
Run 4	0.080	0.068	0.084	0.074	0.025	0.079	0.062	0.076	0.146
MAX	0.080	0.068	0.084	0.074	0.086	0.083	0.080	0.088	0.146
MIN	0.047	0.040	0.016	0.018	0.025	0.005	0.016	0.055	0.053
AVRG	0.059	0.053	0.034	0.032	0.049	0.070	0.044	0.067	0.096
MEDIAN	0.055	0.053	0.028	0.027	0.039	0.076	0.035	0.066	0.089

- Run 4 performs best in desktop mandatory runs without explicit distinction between query types.
 - the summarizer component is very important for 1CLICK task.
- Perform especially better for QA task (30% better than the second best desktop mandatory run)
- Minor improvement by introducing learnt extractor on person type queries