Trainable Tree Distance and an application to Question Categorisation

Martin Emms Department of Computer Science, Trinity College, Dublin Martin.Emms@tcd.ie

Syntactic structures are placed into semantic categories via distances to k nearest neighbours in a pre-categorised set. Variants of tree-distance are used, in particular a stochastic variant. A Viterbi Expectation Maximisation algorithm is proposed via which the parameters of the stochastic model are learned. We show that a 67.7% base-line using standard unit-costs can be improved to 72.5% by cost adaptation.

Tree distance

Standard Edit Distance

a Tai-mapping σ between trees $\mathcal{S}$ and $\mathcal{T}$ is a partial 1-to-1 mapping which

(T1) *preserves left-to-right order* (T2) *preserves ancestry*

Where $\gamma(n)$ is the label of node n, and Σ is all labels, a summed cost can be assigned to a mapping, assuming a cost-table $\mathcal{C}$ size $|\Sigma+1| \times |\Sigma+1|$:

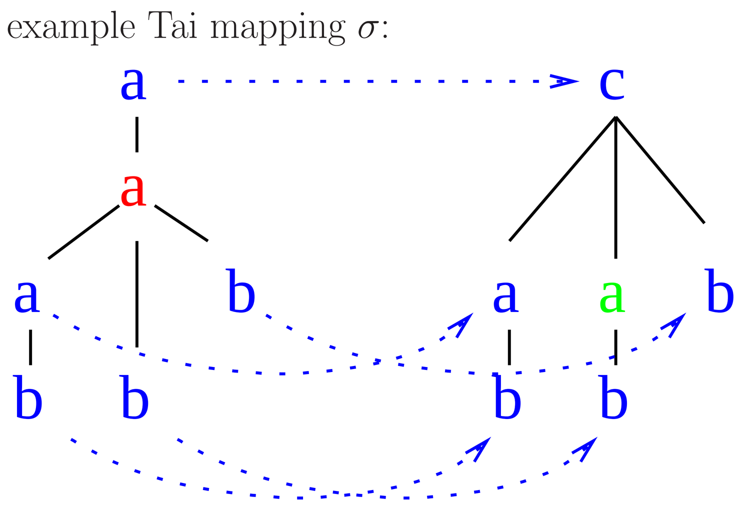
Deletions : $n \in \mathcal{S}, \neg\exists n' \in \mathcal{T}, \langle n, n' \rangle \in \sigma$ $Cost = \mathcal{C}[x][\lambda]$ where $x = \gamma(n)$

Insertions : $n' \in \mathcal{T}, \neg\exists n \in \mathcal{S}, \langle n, n' \rangle \in \sigma$ $Cost = \mathcal{C}[\lambda][y]$ where $y = \gamma(n')$

Swaps/Matches : $n \in \mathcal{S}, n' \in \mathcal{T}, \langle n, n' \rangle \in \sigma$ $Cost = \mathcal{C}[x][y]$ where $x = \gamma(n), y = \gamma(n')$

Definition 0.1 *(Tree- or Tai-distance) between $\mathcal{S}$ and $\mathcal{T}$ is the cost of* **the least-costly Tai mapping** *from $\mathcal{S}$ to $\mathcal{T}$*

example Tai mapping σ:



example cost table:

	λ	a	b	c
λ		•	1	•
a	1	0	•	1
b	•	•	0	•
c	•	•	•	•

cost of σ:

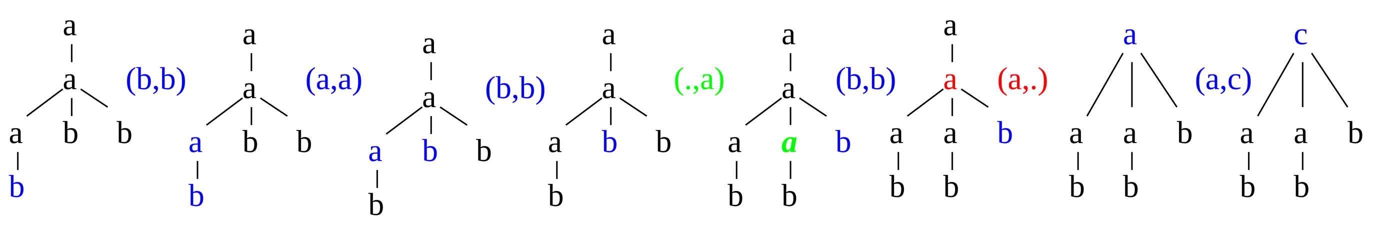
$\mathcal{C}[b][b]$ 0
$\mathcal{C}[a][a]$ 0
$\mathcal{C}[b][b]$ 0
$\mathcal{C}[\lambda][a]$ 1
$\mathcal{C}[b][b]$ 0
$\mathcal{C}[a][\lambda]$ 1
$\mathcal{C}[a][c]$ 1

total = 3

this is also a least cost mapping for this table

Stochastic Edit Distance

A Tai-mapping can also be serialised in a sequence of edit operations, called an edit-script:



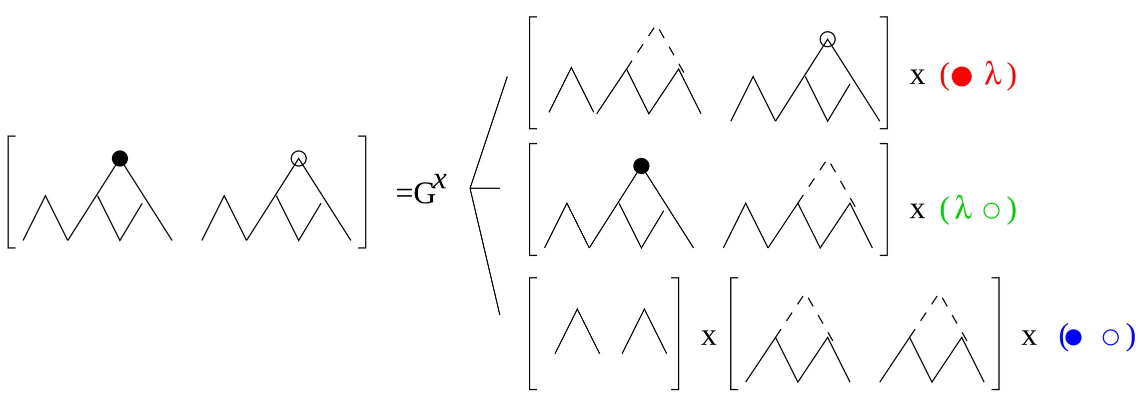
A probability distribution p on edit-script components $e \in (\Sigma \cup \{\lambda\}) \times (\Sigma \cup \{\lambda\})$ can be assumed, and an overall edit-script probability defined as

$$P(e_1 \ldots e_n) = p(e_1) \times \ldots \times p(e_n) \qquad (\text{equiv. } log(P(e_1 \ldots e_n)) = log(p(e_1)) + \ldots + log(p(e_n)))$$

leading to the notions:

Definition 0.2 *(All-paths and Viterbi stochastic Tai distance) $\Delta^A(S,T)$ is the sum of the probabilities of all edit-scripts which represent a Tai-mapping from S to T; $\Delta^V(S,T)$ is the probability of the most probable edit-script*

Algorithms to calculate $\Delta^V(S,T)$ and $\Delta^A(S,T)$ can be based on the following decomposition



For Δ^A, G^A is sum
For Δ^V, G^V is min

Classification via Tree distance

A syntactic structure T can be given a semantic category via its distances to k nearest neighbours in a pre-categorised example set:

```
knn_class(ES, Δ, k;T) {
  let 𝒟 = SORT({(S, Δ(S,T)) | S ∈ ES })
      𝒫 = top(k, 𝒟)
      𝒱 = weighting(𝒫)
  return category with highest vote in 𝒱
}
```

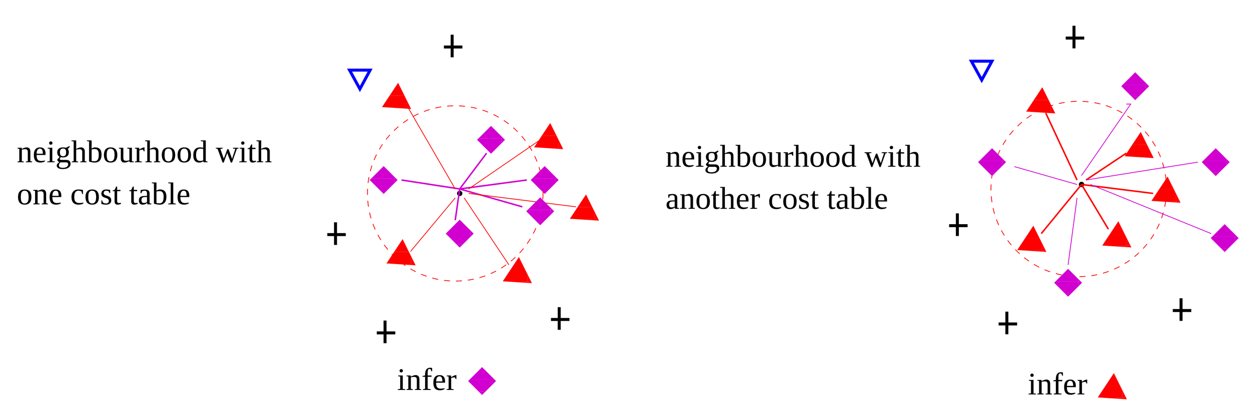
ES is the example set

The *weighting* converts the panel of distance-rated items to weighted votes for their categories.

$vote(C,d) = (d_{max} - d)/(d_{max} - d_{min})$, or 1 if $d_{max} = d_{min}$,

where d_{max} and d_{min} are maximum and minimum distances in the panel.

Different settings for the cost table will give different nearest neighbours and thereby categorisation outcomes, leading to the question of cost-adaptation:



Data for experiments

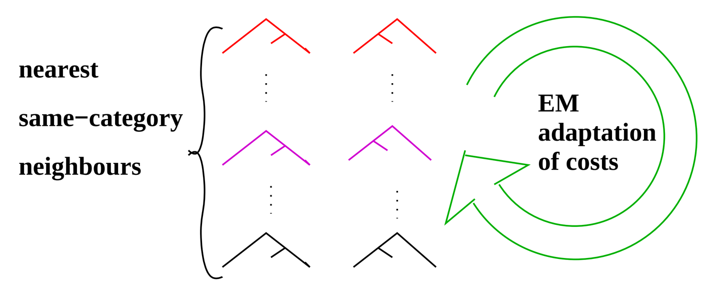
QuestionBank (QB) is a hand-corrected syntactic corpus of questions [3]. A substantical subset of QB comes from a corpus of semantically categorised, syntactically unannotated questions (the CCG corpus from the University of Illinois 2001). From these we created a corpus of 2755 **semantically categorised, syntactically analysed** questions

Cat	Perc	Example
HUM	23.5%	What is the name of the managing director of Apricot Computer ?
ENTY	22.5%	What does the Peugeot company manufacture ?
DESC	19.4%	What did John Hinckley do to impress Jodie Foster ?
NUM	16.7%	When was London 's Docklands Light Railway constructed ?
LOC	16.5%	What country is the biggest producer of tungsten ?
ABBR	1.4%	What is the acronym for the rating system for air conditioner efficiency ?

Experiments were done on 9:1 splits of this data

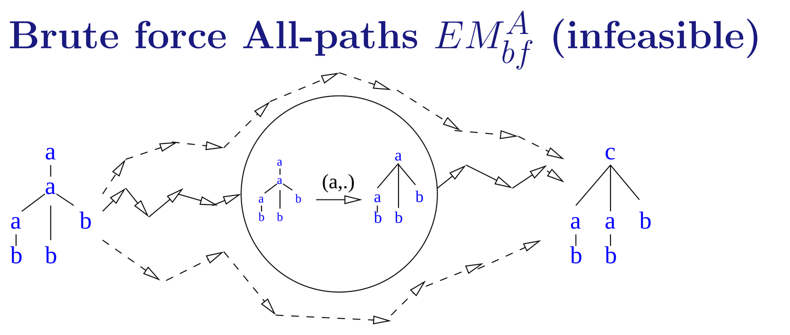
Cost Adaptation via Expectation Maximisation

in scripts between same-category neighbours, intuitively edit-operations should not have uniform probability eg. $P(who/when) << P(state/country)$. We propose to use a corpus of same-category nearest neighbours to adapt costs using an Expectation-Maximisation algorithm.



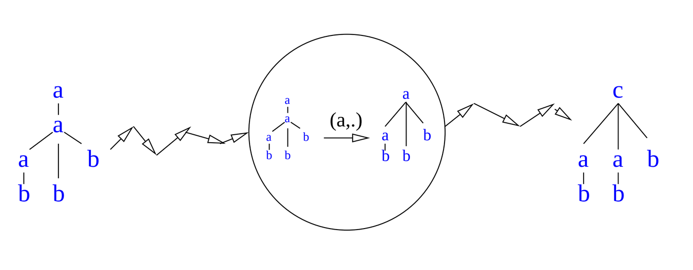
An exponentially expensive algorithm EM^A_{bf} would treat each training pair (S,T) of same-category neighbours as standing for all the edit-scripts $\mathcal{A} : S \mapsto T$, weighting each by its conditional probability, and thereby deriving weighted counts for each op (*see left below*). A *Viterbi variant*, EM^V, approximates this by computing counts from only the best-path $\mathcal{V}$ (*see right below*) . Feasibly implementing EM^A_{bf} is an unsolved problem. [2] contains an incorrect proposal.

Brute force All-paths EM^A_{bf} (infeasible)



$$n_{S,T}(op) = \sum_{\mathcal{A}:S \mapsto T} [\frac{P(\mathcal{A})}{\Delta^A(S,T)} \times \#(op \in \mathcal{A})]$$

Viterbi approximation EM^V (feasible)



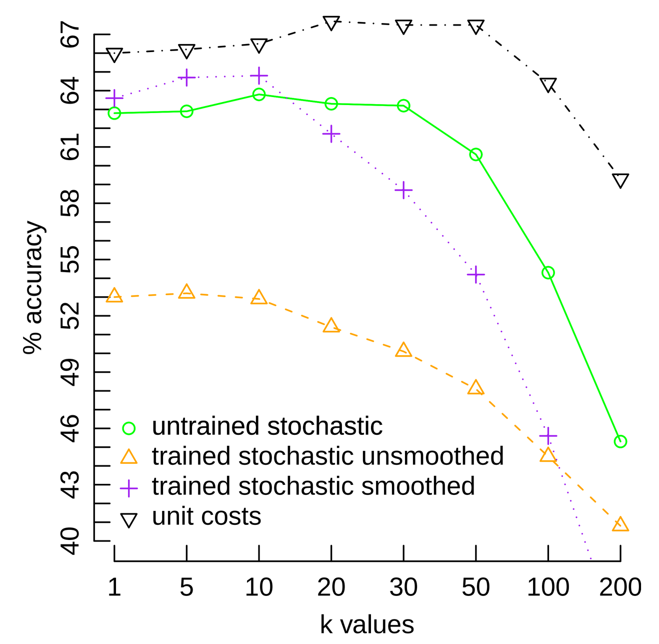
$$n_{(S,T)}(op) = \frac{\Delta^V(S,T)}{\Delta^A(S,T)} \times \#(op \in \mathcal{V})$$

Costs are *initialised* to $\mathcal{C}_u(d)$ where diagonal entries are d times more probable than non-diagonal and costs $\mathcal{C}$ derived by EM^V may be *smoothed* by interpolation with the original $\mathcal{C}_u(d)$ according to $2^{-\mathcal{C}_\lambda[x][y]} = \lambda(2^{-\mathcal{C}[x][y]}) + (1-\lambda)(2^{-\mathcal{C}_u(d)[x][y]})$

Experiments

Experiment One

- unsmoothed EM^V-adapted costs (△,max. 53.2%) worse than initial, stochastic costs (○, max. 63.8%). Testing on the training set though gives 95% accuracy: ⇒ EM^V made the best-scripts connecting the training pairs *too probable, over-fitting the cost table.*
- *smoothing* the adapted costs (+,max. 64.8%) improves over initial costs (○) but is still below unit costs (▽).

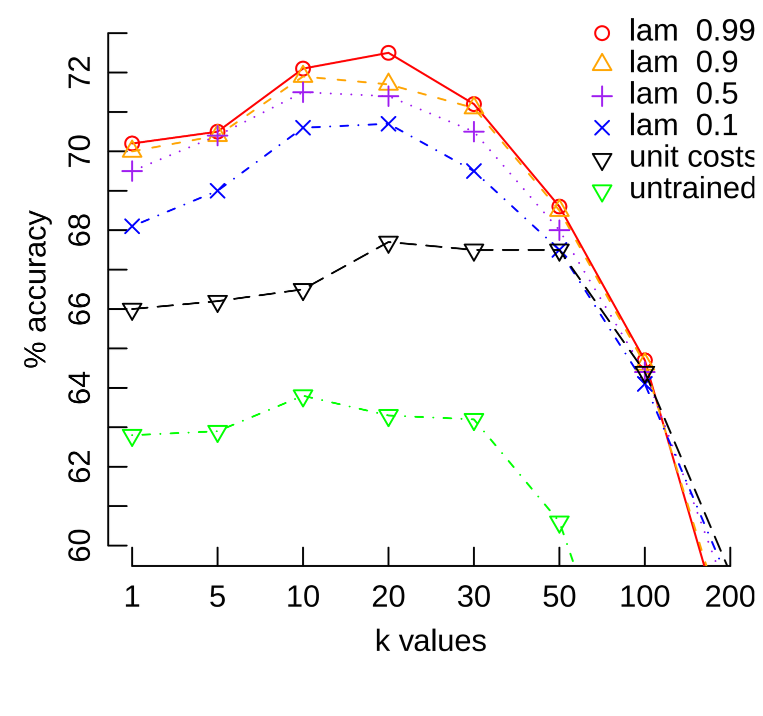


Despite poor categorisation performace, some of the adapted costs seem intuitive. Here is a sample from top 1% of adapted swap costs, which are plausibly discounted relative to others:

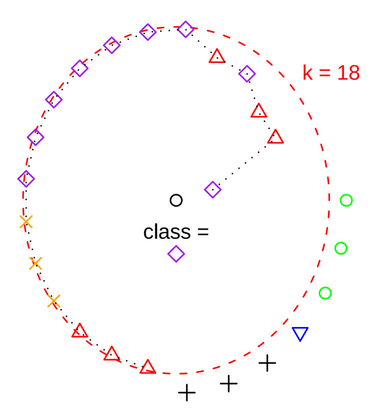
8.50 ? . 9.51 NNS NN 9.78 a the 11.03 was is 12.31 The the 13.60 can do 13.83 many much 13.92 city state

Experiment Two

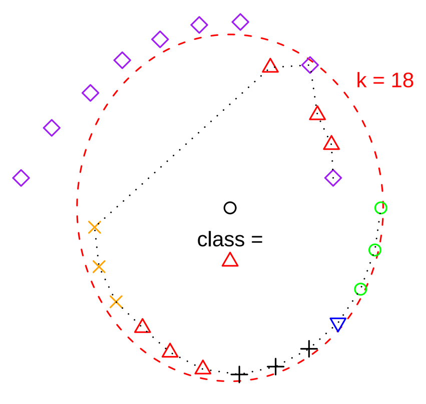
- cost 0 means prob 1 ⇒ a strictly stochastically valid cost table cannot have a zero cost diagonal; perhaps this impedes good categorisation: note the stochastic initialisation $\mathcal{C}_u(3)$ (▽, max. 63.8%) is below unit-costs (▽, max. 67.7%). **We consider outcomes with final step zeroing the diagonal** – this move is also standardly made in cost-adaptation for *string distance* used in duplicate detection [1].
- now with smoothing at varius levels of interpolation ($\lambda \in \{0.99, 0.9, 0.5, 0.1\}$) and with the diagonal zeroed, the EM^V-adapted costs clearly outperform the unit-costs case (▽).
- the best result being 72.5% ($k = 20$, $\lambda = 0.99$), as compared to 67.5% for unit-costs ($k = 20$)



unit costs



adapted costs



plots to the left show an example of misleading neighbours 'migrating' out of the neighbourhood, for an item initially miscategorised as HUM ◇ under unit costs, then correctly categorised as ENTY △ under adapted costs, due in part to learning $P(What/what) >> P(Who/what)$

Comparison and Conclusions

A cost-adapation procedure for $\Delta^V(S,T)$ has been shown to improve the kNN classification performance from 67.7% to 72.5% with adapted costs. If the $SST(S,T)$ tree-kernel 'similarity' is used instead of $\Delta^V(S,T)$ in k-NN, a lower accuracy results: 64% – 69.4%. It remains to compare more closely the $SST(S,T)$ and $\Delta^V(S,T)$ neighbourhoods. However deploying $SST(S,T)$ as a kernel in one-vs-one SVM classification higher accuracies are attainable: 81.3%.

Issues for future work: larger data set, automatically parsed; integration with other lexicon or corpus-based similarity measures; application to other tasks: Question Answering, Entailment Recognition

References

[1] Mikhail Bilenko and Raymond J. Mooney. Adaptive duplicate detection using learnable string similarity measures. KDD 2003

[2] Laurent Boyer, Amaury Habrard, and Marc Sebban. Learning metrics between tree structured data: Application to image recognition. ECML 2007

[3] John Judge. *Adapting and Developing Linguistic Resources for Question Answering.* PhD thesis 2006.