Symbolic answers to an eye-tracking problem

Christine Largeron
EURISE*
largeron@univ-st-etienne.fr

Franck Thollard
EURISE*
thollard@univ-st-etienne.fr

Abstract

We provide in this article experiments made on the eye-tracking challenge proposed by the PASCAL European network. We concentrate here on symbolic approaches mainly based on finite states machines. Our experimental study opens many questions mentioned as a conclusion.

1 Introduction

We address in this paper some experiments made on a shared task proposed by the PASCAL[1] network and which concerns proactive information retrieval [4]. In this task a reader is given a question and 10 sentences, one of them being the correct answer to the question, 4 being relevant and 5 irrelevant. Some information such as the scheduling of the reading or the pupil diameter of the eye of user are stored. During the learning process the machine is given the reading features and the label of the sentences (2 for correct answer, 1 for relevant, and 0 for irrelevant). At evaluation time, the machine is asked to label the sentences. More information on the task together with the data sets can be found at the challenge web page: `http://www.cis.hut.fi/eyechallenge2005/`.

We analyzed the data using different approaches. We first built a graphical interface of the data from which we get a visual rendering of the user behavior. We then used some statistical approaches in order to find relevant features. We then applied decision trees (C5) to handle numerical and categorical features. In order to take into account the behavior of the user, we finally transformed the data in a symbolic form and used syntactic models.

2 Analysis of the data

2.1 Graphical Data Interface: GDI

We built a graphical interface of the data (GDI) – see figure 1 – in order to see what words the user are reading and in which order. On the GDI, we can select a question (or assignment) and a number that allows to tune the time unit. The words of each of the 10 answers are drawn in a color that corresponds to their labels. As the simulation starts, the word being read is colored in a different color, showing the scheduling of the reading.

*EURISE, Jean-Monet University, 42023 Saint-Etienne Cedex 2 France

[1]PASCAL stands for Pattern Analysis, Statistical Modelling and Computational Learning.

Figure 1: Eye tracking Graphical Data Interface

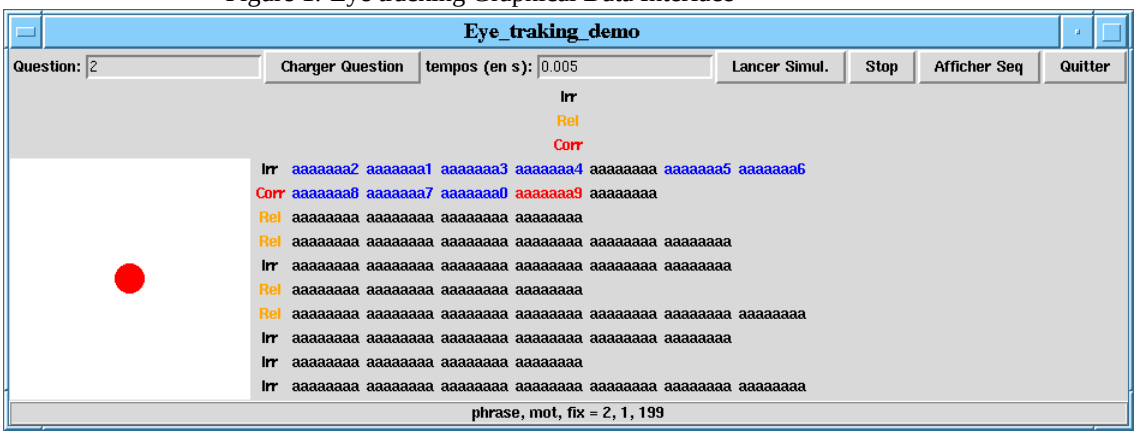


Table 1: Correlation rates

Correlation rates above 0.7	Overall	Label 0	Label 1	Label 2
PrevFixPos – FirstSaccLen	0.794	0.798	0.796	0.785
PupilDiamLag – PupilDiamMax	0.772	0.709	0.721	0.86
MeanFixDur – totalFixDur	0.696	0.711	0.696	0.681
fixcount – firstPassCnt	0.674	0.813	0.778	0.496

On the left hand side of the sentences, a circle is drawn in the color of the label of the word being read; Its size changes according to the pupil diameter.

This GDI allows us to see that the users, almost always, finish the parsing of the 10 answers on the correct one. Labeling the last read sentence as label 2 performs a precision and recall around 92.5% on the validation set.

2.2 Statistical analysis of the data

When facing a new problem, a first natural step could consist in understanding the data. We therefore made a computation of correlation rates and principal component analysis.

2.2.1 The correlation rate

We first computed the correlation rates between the numerical variables. As shown in table 1, the rates were not very high: only very few correlation rates are above .7. The highest value is obtained for prevFixPos and firstSaccLen. But, when we considered only the records corresponding to each label, the rates did not stay constant. For instance the correlation between fixcount and firstPassCnt is only equal to 0.496 for the label 2 when is equal to 0.674 on the training set. It seems then not possible to reduce the number of features by leaving out highly correlated features.

2.2.2 The principal component analysis

We continued our study with Principal Component Analysis (PCA) in order to find out whether there were clusters in the data. We used centered data to preserve the distance between the records instead of normalized data as usually.

Table 2: PCA and C5

PCA	C5	Comp. nb.	Eigen val.	% of var.	cumul. var.
fixcount	fixcount	1	3807987.65	85.249	85.249
firstpasscnt	firstpasscnt	2	465286.091	10.416	95.665
prevfixdur	P1stFixation	3	74205.070	1.661	97.326
firstfixdur	P2stFixation	4	44286.255	.991	98.318
firstpassfixdur	prevfixdur	5	26521.484	.594	98.911
nextfixdur	firstfixdur	6	14507.983	.325	99.236
v11n	firstpassfixdur	7	7774.952	.174	99.410
lastsacclen	nextfixdur	8	7009.023	.157	99.567
prevfixpos	firstSaccLen	9	5245.223	.117	99.685
landingpos	lastsacclen	10	5107.055	.114	99.799
leavingpos	prevfixpos	11	4116.011	9.214E-02	99.891
totalfixdur	landingpos	12	3881.247	8.689E-02	99.978
meanfixdur	leavingpos	13	984.843	2.205E-02	100.00
nregressfrom	totalfixdur	14	.297	6.659E-06	100.00
regresslen	meanfixdur	15	.281	6.283E-06	100.00
regressdur	nregressfrom	16	4.971E-02	1.113E-06	100.00
puplidiammax	regresslen	17	3.716E-02	8.320E-07	100.00
pupildiamlag	nextWordRegress	18	1.602E-02	3.586E-07	100.00
timeprtctg	regressdur	19	4.204E-04	9.411E-09	100.00
	puplidiammax				
	pupildiamlag				
	timeprtctg				

Table 3: Confusion Matrix for the C5 algorithm

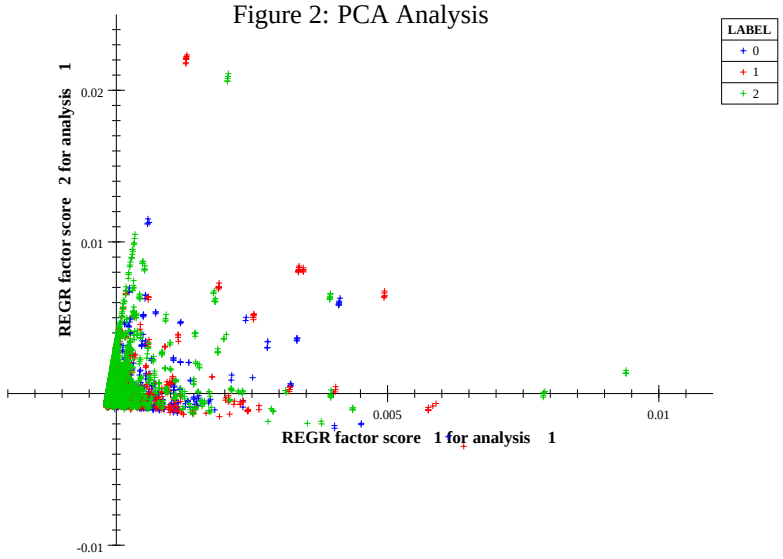
True / Predicted	0	1	2
0	377	196	9
1	255	226	2
2	5	6	138

Table 2 (left, first column) gives the list of the variables used in the PCA. As we can see in table 2, (right) factor 1 accounts for 85.25% of the variance, factor 2 for 10,42% , and so on. The last column contains the cumulative variance extracted. According to the Cattell's criterion [1], we could retain two factors to summarize the data set. Judging from the projection of the training set on the two principal axes, it was not possible to separate the three clusters. The identification of each element by its label on the PCA plot (Figure 2) confirmed this result.

These conclusions lead us to consider a large set of variables, numerical and categorical, and to use C5 classifier designed by Quinlan[2] to handle the features given in Table 2 column 2. Besides that, all the records corresponding to the last sentence read have been excluded. Following the rule deduced from the GDI, the decision for these records is correct answer (label = 2). Results over the evaluation set are reported in Table 3. As expected by the preliminary analysis, results are not extremely high insofar as accuracy is 61.04% on the evaluation and 60.57% on the test set. We thus decided to model the user's behavior.

[2]See `http://www.rulequest.com/see5-info.html` for more details on the C5 algorithm.

Figure 2: PCA Analysis



3 Probabilistic finite state models

The idea of the approach consists in modeling the reading of the user as a path in a finite states machine. We applied the following strategy:

1. discretizing the data,
2. splitting the training set in order to have a learning set for each target label,
3. building three models, one for each label,
4. guessing the label according to each model, and the fact that there is exactly 4 (respectively 5) relevant (respectively irrelevant) sentences in each assignment.

3.1 Discretizing the data

The aim of the discretization is to model the behavior of the user as a string.

We decided to describe an eye movement and its intensity by a pair of characters. We built, by hand, a 9 words vocabulary: B0 B1 B2 E F0 F1 F2 Q0 Q1. Except for the letter E which models the end of the reading, each symbol is composed by two components, a letter indicating an eye movement and a number modeling how important the movement was. B stands for Backward reading, F for Forward reading, and Q for Quitting the sentence.

3.2 Building the models

From this coding we built three multisets of strings (one for each label) of the form:

F0 F0 F0 Q0
F0 F0 F0 Q1
F0 F0 F0 Q1
F0 Q1
F0 F0 Q1
F0 Q1 F0 F1 F1 B0 Q0 B0 E
F0 Q1

Table 4: Perplexity of the models (the lower, the better).
0 – 1 means "learning on train 0 and testing on validation 1"

parameter	0–0	0–1	1–0	1–1
0.05	5.13839	6.6329	5.32001	6.4469
0.01	3.46395	4.69896	3.58998	4.55078
0.002	2.94324	4.68004	4.28921	8.33305
0.005	2.99874	4.64709	3.10137	4.3912
0.0005	8.67323	22.7453	8.94031	21.9032

Table 5: Confusion Matrix for the different approaches

	Automaton model			3-gram model			3-gram - ad-hoc		
True / Predicted	0	1	2	0	1	2	0	1	2
0	438	135	9	421	152	9	381	194	7
1	267	214	2	248	233	2	183	296	4
2	2	9	138	2	9	138	2	9	138
Overall Accuracy	65%			65%			67%		

Since the sets are multisets, we decided to take into account this information by building probabilistic models. We used two kind of models: smoothed trigram and probabilistic automata. Each of these models provides a probability distribution over $\Sigma^\star$, Σ being the vocabulary.

The algorithm for inferring probabilistic automata [3] has a tuning parameter. We usually get the value of the parameter by minimizing the perplexity [2] (or equivalently maximizing the average of the probabilities the inferred automaton provides) on a held out set. Table 4 provides the perplexity obtained by the models on different data. Column 0–1 means: train the model on irrelevant sentences and evaluate it on a held out set of relevant ones.

According to table 4, we decided to use the parameter 0.0005 in order to maximize the margin between label 0 and 1.

3.3 Guessing the label

For we have a quite good rule for label 2 (extracted from the GDI), we decided to first set the label 2 for the sentences on which the user finishes the reading (*i.e.* sentences that contain E in their coding) and then consider a two class problem.

On table 4 we can see that the model built on relevant sentences is not good as it predicts better irrelevant sentences than relevant ones. We thus decided to consider only the model built on irrelevant sentences (*i.e.* sentences labelled 0) and accordingly set label 0 to the 5 more probable sentences according to the model 0 and label 1 to the other ones. The performances of this strategy is given in table 5, left, and performs a global accuracy of 65 % on the validation set. We did the same experiments using a 3-grams model which obtains equivalent performances (65%, table 5, center).

Note that a specific method has been designed by hand by C. de la Higuera for guessing the label given the models. This ad-hoc strategy raises the performances to 67% (table 5 right).

4 Conclusion and further works

In this article, we proposed to use symbolic approaches in order to tackle the eye tracking problem. We identified different steps: building a symbolic coding of the data, inferring syntactic models and guessing the final labels.

We proposed different methods for each step: automatic `vs` hand-made building of the coding, building probabilistic automata `vs` 3-grams, general method for guessing and ad-hoc method. Even if the results are not as bad as compared to the other methods, we now face more questions than answers:

Automatic building of the coding: building the coding automatically is a problem in itself. We tried to build the coding automatically using the rules provided by the C5 algorithm but the preliminary results were very disappointing (*i.e.* $\sim$ 55% of accuracy on the validation set). We thus built the coding by hand keeping in mind the following rules:

- the same string must belong to only one class,
- the vocabulary must be quite small in order to avoid the ”sparse data problem”,
- the sentences of the coding must be quite short,
- the vocabulary must model/select relevant features (e.g. the E symbol that model the ”end of reading”).

In order to optimize the coding itself, it would be good to define an ”off line” quality measure of a coding, that is, in some way, quantifying the above rules.

Quality measure for the inference: as seen before, the best value for the tuning parameter for this task was not the one for which the better model – in term of prediction power– is built. We thus think that a new quality measure is needed in such a case.

Final guess of the label: in the experiments presented, we noted that the results were drastically improved when a consistent labelling is guaranteed (which means, in our case, exactly one sentence is labelled correct, 4 relevant and 5 irrelevant). Moreover, the results can be very different depending on the job done at that step. We think that some more automated work is needed here.

Following one of the anonymous reviewers who ”guess that the strengths of symbolic approach [...] might be simplicity, robustness and speed of implementation”, we would like to continue this work in that direction.

Acknowledgements The authors wish to thank Colin de la Higuera, Thierry Murgue and Jean-Christophe Janodet for fruitful discussions.

References

[1] Cattell, R. (1966) The scree test for the number of factors. *Multivariate Behavioral Research*, 1.

[2] Goodman, J. (2001) *A bit of Progress in Language Modeling*, Technical report MSR-TR-2001-72.

[3] Thollard, F. (2001) *Improving Probabilistic Grammatical Inference Core Algorithms with Post-processing Techniques*, pp 561-568, ICML’2001, Williams/Morgan Kauffman

[4] Salojrvi, J & Kojo, I. & Simola, J. & and Kaski, S.: Can relevance be inferred from eye movements in information retrieval? In (WSOM’03), Hibikino, Kitakyushu, Japan, 2003. pp. 261-266.