Comparing Multi-label Classification with Reinforcement Learning for Summarisation of Time-series Data

Dimitra Gkatzia, Helen Hastie, and Oliver Lemon
School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh
{dg106, h.hastie, o.lemon}@hw.ac.uk

Abstract

We present a novel approach for automatic report generation from time-series data, in the context of student feedback generation. Our proposed methodology treats content selection as a multi-label (ML) classification problem, which takes as input time-series data and outputs a set of templates, while capturing the dependencies between selected templates. We show that this method generates output closer to the feedback that lecturers actually generated, achieving 3.5% higher accuracy and 15% higher F-score than multiple simple classifiers that keep a history of selected templates. Furthermore, we compare a ML classifier with a Reinforcement Learning (RL) approach in simulation and using ratings from real student users. We show that the different methods have different benefits, with ML being more accurate for predicting what was seen in the training data, whereas RL is more exploratory and slightly preferred by the students.

1 Introduction

Summarisation of time-series data refers to the task of automatically generating text from variables whose values change over time. We consider the task of automatically generating feedback summaries for students describing their performance during the lab of a Computer Science module over the semester. Students' learning can be influenced by many variables, such as difficulty of the material (Person et al., 1995), other deadlines (Craig et al., 2004), attendance in lectures (Ames, 1992), etc. These variables have two important qualities. Firstly, they change over time, and secondly they can be dependent on or independent of each other. Therefore, when generating feedback, we need to take into account all variables simultaneously in order to capture potential dependencies and provide more effective and useful feedback that is relevant to the students.

In this work, we concentrate on content selection which is the task of choosing what to say, i.e. what information is to be included in a report (Reiter and Dale, 2000). Content selection decisions based on trends in time-series data determine the selection of the useful and important variables, which we refer to here as *factors*, that should be conveyed in a summary. The decisions of factor selection can be influenced by other factors that their values are correlated with; can be based on the appearance or absence of other factors in the summary; and can be based on the factors' behaviour over time. Moreover, some factors may have to be discussed together in order to achieve some communicative goal, for instance, a teacher might want to refer to student's marks as a motivation for increasing the number of hours studied.

We frame content selection as a simple classification task: given a set of time-series data, decide for each template whether it should be included in a summary or not. In this paper, with the term 'template' we refer to a quadruple consisting of an *id*, a *factor* (bottom left of Table 1), a *reference type* (trend, weeks, average, other) and *surface text*. However, simple classification assumes that the templates are independent of each other, thus the decision for each template is taken in isolation from the others, which is not appropriate for our domain. In order to capture the dependencies in the context, multiple simple classifiers can make the decisions for each template iteratively. After each iteration, the feature space grows by 1 feature, in order to include the history of the previous template decisions. Here, we propose an alternative method that tackles the challenge of interdependent data by using multi-label (ML) classification, which is efficient in taking data dependencies

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 1231–1240,
Baltimore, Maryland, USA, June 23-25 2014.

Raw Data

factors	week 2	week 3	...	week 10
marks	5	4	...	5
hours_studied	1	2	...	3
...	...	...	...	...

Trends from Data

factors	trend
(1) marks (M)	trend_other
(2) hours_studied (HS)	trend_increasing
(3) understandability (Und)	trend_decreasing
(4) difficulty (Diff)	trend_decreasing
(5) deadlines (DL)	trend_increasing
(6) health_issues (HI)	trend_other
(7) personal_issues (PI)	trend_decreasing
(8) lectures_attended (LA)	trend_other
(9) revision (R)	trend_decreasing

Summary

Your overall performance **was excellent** during the semester. Keep up the good work and maybe try some more challenging exercises. Your attendance was **varying** over the semester. Have a think about how to use time in lectures to improve your understanding of the material. You spent **2 hours studying the lecture material on average**. You should dedicate more time to study. You seem to find the material **easier to understand compared to the beginning of the semester**. Keep up the good work! You revised **part of** the learning material. Have a think whether revising has improved your performance.

Table 1: The table on the top left shows an example of the time-series raw data for feedback generation. The table on the bottom left shows an example of described trends. The box on the right presents a target summary (target summaries have been constructed by teaching staff).

into account and generating a set of labels (in our case templates) simultaneously (Tsoumakas et al., 2010). ML classification requires no history, i.e. does not keep track of previous decisions, and thus has a smaller feature space.

Our contributions to the field are as follows: we present a novel and efficient method for tackling the challenge of content selection using a ML classification approach; we applied this method to the domain of feedback summarisation; we present a comparison with an optimisation technique (Reinforcement Learning), and we discuss the similarities and differences between the two methods.

In the next section, we refer to the related work on Natural Language Generation from time-series data and on Content Selection. In Section 4.2, we describe our approach and we carry out a comparison with simple classification methods. In Section 5, we present the evaluation setup and in Section 6 we discuss the results, obtained in simulation and with real students. Finally, in Section 8, directions for future work are discussed.

2 Related Work

Natural Language Generation from time-series data has been investigated for various tasks such as weather forecast generation (Belz and Kow, 2010; Angeli et al., 2010; Sripada et al., 2004), report generation from clinical data (Hunter et al., 2011; Gatt et al., 2009), narrative to assist children with communication needs (Black et al., 2010) and audiovisual debrief generation from sensor data from Autonomous Underwater Vehicles missions (Johnson and Lane, 2011).

The important tasks of time-series data summarisation systems are *content selection* (what to say), *surface realisation* (how to say it) and *information presentation* (Document Planning, Ordering, etc.). In this work, we concentrate on content selection. Previous methods for content selection include Reinforcement Learning (Rieser et al., 2010); multi-objective optimisation (Gkatzia et al., 2014); Gricean Maxims (Sripada et al., 2003); Integer Linear Programming (Lampouras and Androutsopoulos, 2013); collective content selection (Barzilay and Lapata, 2004); interest scores assigned to content (Androutsopoulos et al., 2013); a combination of statistical and template-based approaches to NLG (Kondadadi et al., 2013); statistical acquisition of rules (Duboue and McKeown, 2003) and the Hidden Markov model approach for Content Selection and ordering (Barzilay and Lee, 2004).

Collective content selection (Barzilay and Lapata, 2004) is similar to our proposed method in that it is a classification task that predicts the templates from the same instance simultaneously. The difference between the two methods lies in that the

collective content selection requires the consideration of an individual preference score (which is defined as the preference of the entity to be selected or omitted, and it is based on the values of entity attributes and is computed using a boosting algorithm) and the identification of links between the entities with similar labels. In contrast, ML classification does not need the computation of links between the data and the templates. ML classification can also apply to other problems whose features are correlated, such as text classification (Madjarov et al., 2012), when an aligned dataset is provided.

ML classification algorithms have been divided into three categories: algorithm adaptation methods, problem transformation and ensemble methods (Tsoumakas and Katakis, 2007; Madjarov et al., 2012). Algorithm adaptation approaches (Tsoumakas et al., 2010) extend simple classification methods to handle ML data. For example, the k-nearest neighbour algorithm is extended to ML-kNN by Zhang and Zhou (2007). ML-kNN identifies for each new instance its k nearest neighbours in the training set and then it predicts the label set by utilising the maximum a posteriori principle according to statistical information derived from the label sets of the k neighbours. Problem transformation approaches (Tsoumakas and Katakis, 2007) transform the ML classification task into one or more simple classification tasks. Ensemble methods (Tsoumakas et al., 2010) are algorithms that use ensembles to perform ML learning and they are based on problem transformation or algorithm adaptation methods. In this paper, we applied RAkEL (Random k-labelsets) (Tsoumakas et al., 2010): an ensemble problem transformation method, which constructs an ensemble of simple-label classifiers, where each one deals with a random subset of the labels.

Finally, our domain for feedback generation is motivated by previous studies (Law et al., 2005; van den Meulen et al., 2010) who show that text summaries are more effective in decision making than graphs therefore it is advantageous to provide a summary over showing users the raw data graphically. In addition, feedback summarisation from time-series data can be applied to the field of Intelligent Tutoring Systems (Gross et al., 2012).

3 Data

The dataset consists of 37 instances referring to the activities of 26 students. For a few students there is more than 1 instance. An example of one such instance is presented in Table 1. Each instance includes time-series information about the student's learning habits and the selected templates that lecturers used to provide feedback to this student. The time-series information includes for each week of the semester: (1) the marks achieved at the lab; (2) the hours that the student spent studying; (3) the understandability of the material; (4) the difficulty of the lab exercises as assessed by the student; (5) the number of other deadlines that the student had that week; (6) health issues; (7) personal issues; (8) the number of lectures attended; and (9) the amount of revision that the student had performed. The templates describe these factors in four different ways:

1. <**trend**>**:** referring to the trend of a factor over the semester (e.g. "Your performance *was increasing*..."),
2. <**weeks**>**:** explicitly describing the factor value at specific weeks (e.g. "In *weeks 2, 3 and 9*..."),
3. <**average**>**:** considering the average of a factor value (e.g. "You dedicated *1.5 hours studying on average*..."), and
4. <**other**>**:** mentioning other relevant information (e.g. "*Revising material will improve your performance*").

For the corpus creation, 11 lecturers selected the content to be conveyed in a summary, given the set of raw data (Gkatzia et al., 2013). As a result, for the same student there are various summaries provided by the different experts. This characteristic of the dataset, that each instance is associated with more than one solution, additionally motivates the use of multi-label classification, which is concerned with learning from examples, where each example is associated with multiple labels.

Our analysis of the dataset showed that there are significant correlations between the factors, for example, the number of lectures attended (LA) correlates with the student's understanding of the material (Und), see Table 2. As we will discuss further in Section 5.1, content decisions are influenced by the previously generated content, for example, if the lecturer has previously mentioned health_issues, mentioning hours_studied has a high probability of also being mentioned.

Factor	(1) M	(2) HS	(3) Und	(4) Diff	(5) DL	(6) HI	(7) PI	(8) LA	(9) R
(1) M	1*	0.52*	0.44*	-0.53*	-0.31	-0.30	-0.36*	0.44*	0.16
(2) HS	0.52*	1*	0.23	-0.09	-0.11	0.11	-0.29	0.32	0.47*
(3) Und	0.44*	0.23	1*	-0.54*	0.03	-0.26	0.12	0.60*	0.32
(4) Diff	-0.53*	-0.09	-0.54*	1*	0.16	-0.06	0.03	-0.19	0.14
(5) DL	-0.31	-0.11	0.03	0.16	1*	0.26	0.24	-0.44*	0.14
(6) HI	-0.30	-0.11	-0.26	-0.06	0.26	1*	0.27	-0.50*	0.15
(7) PI	-0.36*	-0.29	0.12	0.03	0.24	0.27	1*	-0.46*	0.34*
(8) LA	0.44*	0.32	0.60*	-0.19	-0.44*	-0.50*	-0.46*	1*	-0.12
(9) R	0.16	0.47*	0.03	0.14	0.14	0.15	0.34*	-0.12	1*

Table 2: The table presents the Pearson's correlation coefficients of the factors (* means $p<0.05$).

4 Methodology

In this section, the content selection task and the suggested multi-label classification approach are presented. The development and evaluation of the time-series generation system follows the following pipeline (Gkatzia et al., 2013):

1. Time-Series data collection from students
2. Template construction by Learning and Teaching (L&T) expert
3. Feedback summaries constructed by lecturers; random summaries rated by lecturers
4. Development of time-series generation systems (Section 4.2, Section 5.3): ML system, RL system, Rule-based and Random system
5. Evaluation: (Section 5)
   - Offline evaluation (Accuracy and Reward)
   - Online evaluation (Subjective Ratings)

4.1 The Content Selection Task

Our learning task is formed as follows: given a set of 9 time-series factors, select the content that is most appropriate to be included in a summary. Content is regarded as labels (each template represents a label) and thus the task can be thought of as a classification problem. As mentioned, there are 4 ways to refer to a factor: (1) describing the trend, (2) describing what happened in every time stamp, (3) mentioning the average and (4) making another general statement. Overall, for all factors there are 29 different templates[1]. An example of the input data is shown in Table 1. There are two decisions that need to be made: (1) whether to talk about a factor and (2) in which way to refer to it. Instead of dealing with this task in a hierarchical way, where the algorithm will first learn whether to talk about a factor and then to decide how to refer to it, we transformed the task in order to reduce the learning steps. Therefore, classification can reduce the decision workload by deciding either in which way to talk about it, or not to talk about a factor at all.

[1]There are fewer than 36 templates, because for some factors there are less than 4 possible ways of referring to them.

4.2 The Multi-label Classification Approach

Traditional single-label classification is the task of identifying which label one new observation is associated with, by choosing from a set of labels L (Tsoumakas et al., 2010). Multi-label classification is the task of associating an observation with a set of labels $Y \subseteq L$ (Tsoumakas et al., 2010).

One set of factor values can result in various sets of templates as interpreted by the different experts. A ML classifier is able to make decisions for all templates simultaneously and capture these differences. The RAndom k-labELsets (RAkEL) (Tsoumakas et al., 2010) was applied in order to perform ML classification. RAkEL is based on Label Powerset (LP), a problem transformation method (Tsoumakas et al., 2010). LP benefits from taking into consideration label correlations, but does not perform well when trained with few examples as in our case (Tsoumakas et al., 2010). RAkEL overcomes this limitation by constructing a set of LP classifiers, which are trained with different random subsets of the set of labels (Tsoumakas et al., 2010).

The LP method transforms the ML task, into one single-label multi-class classification task, where the possible set of predicted variables for the transformed class is the powerset of labels present in the original dataset. For instance, the set of labels $L = \{temp_0, temp_1, ...temp_{28}\}$ could be transformed to $\{temp_{0,1,2}, temp_{28,3,17,}...\}$. This algorithm does not perform well when considering a large number of labels, due to the fact that the label space grows exponentially (Tsoumakas

Classifier	Accuracy (10-fold)	Precision	Recall	F score
Decision Tree (no history)	*75.95%	67.56	75.96	67.87
Decision Tree (with predicted history)	**73.43%	65.49	72.05	70.95
Decision Tree (with real history)	**78.09%	74.51	78.11	75.54
Majority-class (single label)	**72.02%	61.73	77.37	68.21
RAkEL (multi-label) (no history)	76.95%	**85.08**	**85.94**	**85.50**

Table 3: Average, precision, recall and F-score of the different classification methods (T-test, * denotes significance with $p<0.05$ and ** significance with $p<0.01$, when comparing each result to RAkEL).

et al., 2010). RAkEL tackles this problem by constructing an ensemble of LP classifiers and training each one on a different random subset of the set of labels (Tsoumakas et al., 2010).

4.2.1 The Production Phase of RAkEL

The algorithm was implemented using the MULAN Open Source Java library (Tsoumakas et al., 2011), which is based on WEKA (Witten and Frank, 2005). The algorithm works in two phases:

1. the production of an ensemble of LP algorithms, and
2. the combination of the LP algorithms.

RAkEL takes as input the following parameters: (1) the numbers of iterations m (which is developer specified and denotes the number of models that the algorithm will produce), (2) the size of labelset k (which is also developer specified), (3) the set of labels L, and (4) the training set D. During the initial phase it outputs an ensemble of LP classifiers and the corresponding k-labelsets. A pseudocode for the production phase is shown below:

Algorithm 1 RAkEL production phase

```
1: Input: iterations m, k labelsets,
          labels L, training data D

2: for i=0 to m
3:       Select random k-labelset from L
4:       Train an LP on D
5:       Add LP to ensemble
6: end for

7: Output: the ensemble of LPs
       with corresponding k-labelsets
```

4.2.2 The Combination Phase

During the combination phase, the algorithm takes as input the results of the production phase, i.e. the ensemble of LPs with the corresponding k-labelsets, the set of labels L, and the new instance x and it outputs the result vector of predicted labels for instance x. During run time, RAkEL estimates the average decision for each label in L and if the average is greater than a threshold t (determined by the developer) it includes the label in the predicted labelset. We used the standard parameter values of t, k and m (t = 0.5, k = 3 and m equals to 58 (2*29 templates)). In future, we could perform parameter optimisation by using a technique similar to (Gabsdil and Lemon, 2004).

5 Evaluation

Firstly, we performed a preliminary evaluation on classification methods, comparing our proposed ML classification with multiple iterated classification approaches. The summaries generated by the ML classification system are then compared with the output of a RL system and two baseline systems in simulation and with real students.

5.1 Comparison with Simple Classification

We compared the RAkEL algorithm with single-label (SL) classification. Different SL classifiers were trained using WEKA: JRip, Decision Trees, Naive Bayes, k-nearest neighbour, logistic regression, multi-layer perceptron and support vector machines. It was found out that Decision Trees achieved on average 3% higher accuracy. We, therefore, went on to use Decision Trees that use generation history in three ways.

Firstly, for **Decision Tree (no history)**, 29 decision-tree classifiers were trained, one for each template. The input of these classifiers were the 9 factors and each classifier was trained in order to decide whether to include a specific template or not. This method did not take into account other selected templates – it was only based on the time-series data.

Secondly, for **Decision Tree (with predicted history)**, 29 classifiers were also trained, but this time the input included the previous decisions made by the previous classifiers (i.e. the history)

as well as the set of time-series data in order to emulate the dependencies in the dataset. For instance, classifier n was trained using the data from the 9 factors and the template decisions for templates 0 to $n-1$.

Thirdly, for **Decision Tree (with real history)**, the real, expert values were used rather than the predicted ones in the history. The above-mentioned classifiers are compared with, the **Majority-class (single label)** baseline, which labels each instance with the most frequent template.

The accuracy, the weighted precision, the weighted recall, and the weighted F-score of the classifiers are shown in Table 3. It was found that in 10-fold cross validation RAkEL performs significantly better in all these automatic measures (accuracy = 76.95%, F-score = 85.50%). Remarkably, ML achieves more than 10% higher F-score than the other methods (Table 3). The average accuracy of the single-label classifiers is 75.95% (10-fold validation), compared to 73.43% of classification with history. The reduced accuracy of the classification with predicted history is due to the error in the predicted values. In this method, at every step, the predicted outcome was used including the incorrect decisions that the classifier made. The upper-bound accuracy is 78.09% calculated by using the expert previous decisions and not the potentially erroneous predicted decisions. This result is indicative of the significance of the relations between the factors showing that the predicted decisions are dependent due to existing correlations as discussed in Section 1, therefore the system should not take these decisions independently. ML classification performs better because it does take into account these correlations and dependencies in the data.

5.2 The Reinforcement Learning System

Reinforcement Learning (RL) is a machine learning technique that defines how an agent learns to take optimal actions so as to maximise a cumulative reward (Sutton and Barto, 1998). Content selection is seen as a Markov Decision problem and the goal of the agent is to learn to take the sequence of actions that leads to optimal content selection. The Temporal Difference learning method was used to train an agent for content selection.

Actions and States: The state consists of the time-series data and the selected templates. In order to explore the state space the agent selects a factor (e.g. marks, deadlines etc.) and then decides whether to talk about it or not.

Reward Function: The reward function reflects the lecturers' preferences on summaries and is derived through linear regression analysis of a dataset containing lecturer constructed summaries and ratings of randomly generated summaries. Specifically, it is the following cumulative multivariate function:

$$Reward = a + \sum_{i=1}^{n} b_i * x_i + c * length$$

where $X = \{x_1, x_2, ..., x_n\}$ describes the combinations of the data trends observed in the time-series data and a particular template. a, b and c are the regression coefficients, and their values vary from -99 to 221. The value of x_i is given by the function:

$$x_i = \begin{cases} 1, & \text{the combination of a factor trend} \\ & \text{and a template type is included} \\ & \text{in a summary} \\ 0, & \text{if not.} \end{cases}$$

The RL system differs from the classification system in the way it performs content selection. In the training phase, the agent selects a factor and then decides whether to talk about it or not. If the agent decides to refer to a factor, the template is selected in a deterministic way, i.e. from the available templates it selects the template that results in higher expected cumulative future reward.

5.3 The Baseline Systems

We compared the ML system and the RL system with two baselines described below by measuring the accuracy of their outputs, the reward achieved by the reward function used for the RL system, and finally we also performed evaluation with student users. In order to reduce the confounding variables, we kept the ordering of content in all systems the same, by adopting the ordering of the rule-based system. The baselines are as follows:

1. Rule-based System: generates summaries based on Content Selection rules derived by working with a L&T expert and a student (Gkatzia et al., 2013).

2. Random System: initially, selects a factor randomly and then selects a template randomly, until it makes decisions for all factors.

Time-Series Summarisation Systems	Accuracy	Reward	Rating Mode (mean)	Data Source
Multi-label Classification	**85%**	65.4	7 (6.24)	Lecturers' constructed summaries
Reinforcement Learning	**66%	**243.82**	**8 (6.54)**	Lecturers' ratings & summaries
Rule-based	**65%	107.77	7, 8 (5.86)	L&T expert
Random	**45.2%	43.29	*2 (*4.37)	Random

Table 4: Accuracy, average rewards (based on lecturers' preferences) and averages of the means of the student ratings. Accuracy significance (Z-test) with RAkEL at $p<0.05$ is indicated as * and at $p<0.01$ as **. Student ratings significance (Mann Whitney U test) with RAkEL at $p<0.05$ is indicated as *.

6 Results

Each of the four systems described above generated 26 feedback summaries corresponding to the 26 student profiles. These summaries were evaluated in simulation and with real student users.

6.1 Results in Simulation

Table 4 presents the accuracy, reward, and mode of student rating of each algorithm when used to generate the 26 summaries. Accuracy was estimated as the proportion of the correctly classified templates to the population of templates. In order to have a more objective view on the results, the score achieved by each algorithm using the reward function was also calculated. ML classification achieved significantly higher accuracy, which was expected as it is a supervised learning method. The rule-based system and the RL system have lower accuracy compared to the ML system. There is evidently a mismatch between the rules and the test-set; the content selection rules are based on heuristics provided by a L&T Expert rather than by the same pool of lecturers that created the test-set. On the contrary, the RL is trained to optimise the selected content and not to replicate the existing lecturer summaries, hence there is a difference in accuracy.

Accuracy measures how similar the generated output is to the gold standard, whereas the reward function calculates a score regarding how good the output is, given an objective function. RL is trained to optimise for this function, and therefore it achieves higher reward, whereas ML is trained to learn by examples, therefore it produces output closer to the gold standard (lecturer's produced summaries). RL uses exploration and exploitation to discover combinations of content that result in higher reward. The reward represents predicted ratings that lecturers would give to the summary. The reward for the lecturers' produced summaries is 124.62 and for the ML method is 107.77. The ML classification system performed worse than this gold standard in terms of reward, which is expected given the error in predictions (supervised methods learn to reproduce the gold standard). Moreover, each decision is rewarded with a different value as some combinations of factors and templates have greater or negative regression coefficients. For instance, the combination of the factors "deadlines" and the template that corresponds to <weeks> is rewarded with 57. On the other hand, when mentioning the <average> difficulty the summary is "punished" with -81 (see description of the reward function in Section 5.2). Consequently, a single poor decision in the ML classification can result in much less reward.

6.2 Subjective Results with Students

37 first year computer science students participated in the study. Each participant was shown a graphical representation of the time-series data of one student and four different summaries generated by the four systems (see Figure 1). The order of the presented summaries was randomised. They were asked to rate each feedback summary on a 10-point rating scale in response to the following statement: "Imagine you are the following student. How would you evaluate the following feedback summaries from 1 to 10?", where 10 corresponds to the most preferred summary and 1 to the least preferred.

The difference in ratings between the ML classification system, the RL system and the Rule-based system is not significant (see Mode (mean) in Table 4, $p>0.05$). However, there is a trend towards the RL system. The classification method reduces the generation steps, by making the decision of the factor selection and the template selection jointly. Moreover, the training time for the classification method is faster (a couple of seconds compared to over an hour). Finally, the student

Figure 1: The Figure show the evaluation setup. Students were presenting with the data in a graphical way and then they were asked to evaluate each summary in a 10-point Rating scale. Summaries displayed from left to right: ML system, RL, rule-based and random.

significantly prefer all the systems over the random.

7 Summary

We have shown that ML classification for summarisation of our time-series data has an accuracy of 76.95% and that this approach significantly outperforms other classification methods as it is able to capture dependencies in the data when making content selection decisions. ML classification was also directly compared to a RL method. It was found that although ML classification is almost 20% more accurate than RL, both methods perform comparably when rated by humans. This may be due to the fact that the RL optimisation method is able to provide more varied responses over time rather than just emulating the training data as with standard supervised learning approaches. Foster (2008) found similar results when performing a study on generation of emphatic facial displays. A previous study by Belz and Reiter (2006) has demonstrated that automatic metrics can correlate highly with human ratings if the training dataset is of high quality. In our study, the human ratings correlate well to the average scores achieved by the reward function. However, the human ratings do not correlate well to the accuracy scores. It is interesting that the two methods that score differently on various automatic metrics, such as accuracy, reward, precision, recall and F-score, are evaluated similarly by users.

The comparison shows that each method can serve different goals. Multi-label classification generates output closer to gold standard whereas RL can optimise the output according to a reward function. ML classification could be used when the goal of the generation is to replicate phenomena seen in the dataset, because it achieves high accuracy, precision and recall. However, optimisation methods can be more flexible, provide more varied output and can be trained for different goals, e.g. for capturing preferences of different users.

8 Future Work

For this initial experiment, we evaluated with students and not with lecturers, since the students are the recipients of feedback. In future, we plan to evaluate with students' own data under real circumstances as well as with ratings from lecturers. Moreover, we plan to utilise the results from this student evaluation in order to train an optimisation algorithm to perform summarisation according to students' preferences. In this case, optimisation would be the preferred method as it would not be appropriate to collect gold standard data from students. In fact, it would be of interest to investigate multi-objective optimisation techniques that can balance the needs of the lecturers to convey important content to the satisfaction of students.

9 Acknowledgements

The research leading to this work has received funding from the EC's FP7 programme: (FP7/2011-14) under grant agreement no. 248765 (Help4Mood).

References

Carole Ames. 1992. Classrooms: Goals, structures, and student motivation. *Journal of Educational Psychology*, 84(3):261–71.

Ion Androutsopoulos, Gerasimos Lampouras, and Dimitrios Galanis. 2013. Generating natural language descriptions from owl ontologies: the natural owl system. *Atrificial Intelligence Research*, 48:671–715.

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Regina Barzilay and Mirella Lapata. 2004. Collective content selection for concept-to-text generation. In *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP)*.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*.

Anja Belz and Eric Kow. 2010. Extracting parallel fragments from comparable corpora for data-to-text generation. In *6th International Natural Language Generation Conference (INLG)*.

Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of nlg systems. In *11th Conference of the European Chapter of the Association for Computational Linguistics (ACL)*.

Rolf Black, Joe Reddington, Ehud Reiter, Nava Tintarev, and Annalu Waller. 2010. Using NLG and sensors to support personal narrative for children with complex communication needs. In *NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies*.

Scotty D. Craig, Arthur C. Graesser, Jeremiah Sullins, and Barry Gholson. 2004. Affect and learning: an exploratory look into the role of affect in learning with autotutor. *Journal of Educational Media*, 29:241–250.

Pable Duboue and K.R. McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (EMNLP)*.

Mary Ellen Foster. 2008. Automated metrics that agree with human judgements on generated output for an embodied conversational agent. In *5th International Natural Language Generation Conference (INLG)*.

Malte Gabsdil and Oliver Lemon. 2004. Combining acoustic and pragmatic features to predict recognition performance in spoken dialogue systems. In *42nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Albert Gatt, Francois Portet, Ehud Reiter, James Hunter, Saad Mahamood, Wendy Moncur, and Somayajulu Sripada. 2009. From data to text in the neonatal intensive care unit: Using NLG technology for decision support and information management. *AI Communications*, 22: 153-186.

Dimitra Gkatzia, Helen Hastie, Srinivasan Janarthanam, and Oliver Lemon. 2013. Generating student feedback from time-series data using Reinforcement Learning. In *14th European Workshop in Natural Language Generation (ENLG)*.

Dimitra Gkatzia, Helen Hastie, and Oliver Lemon. 2014. Finding Middle Ground? Multi-objective Natural Language Generation from time-series data. In *14th Conference of the European Chapter of the Association for Computational Linguistics (EACL) (to appear)*.

Sebastian Gross, Bassam Mokbel, Barbara Hammer, and Niels Pinkwart. 2012. Feedback provision strategies in intelligent tutoring systems based on clustered solution spaces. In J. Desel, J. M. Haake, and C. Spannagel, editors, *Tagungsband der 10. e-Learning Fachtagung Informatik (DeLFI)*, number P-207 in GI Lecture Notes in Informatics, pages 27–38. GI.

Jim Hunter, Yvonne Freer, Albert Gatt, Yaji Sripada, Cindy Sykes, and D Westwater. 2011. Bt-nurse: Computer generation of natural language shift summaries from complex heterogeneous medical data. *American Medical Informatics Association*, 18:621-624.

Nicholas Johnson and David Lane. 2011. Narrative monologue as a first step towards advanced mission debrief for AUV operator situational awareness. In *15th International Conference on Advanced Robotics*.

Ravi Kondadadi, Blake Howald, and Frank Schilder. 2013. A statistical nlg framework for aggregated planning and realization. In *51st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Gerasimos Lampouras and Ion Androutsopoulos. 2013. Using integer linear programming in concept-to-text generation to produce more compact texts. In *51st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Anna S. Law, Yvonne Freer, Jim Hunter, Robert H. Logie, Neil McIntosh, and John Quinn. 2005. A comparison of graphical and textual presentations of time series data to support medical decision making in the neonatal intensive care unit. *Journal of Clinical Monitoring and Computing*, pages 19: 183–194.

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Saso Dzeroski. 2012. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104.

Natalie K. Person, Roger J. Kreuz, Rolf A. Zwaan, and Arthur C. Graesser. 1995. Pragmatics and pedagogy: Conversational rules and politeness strategies may inhibit effective tutoring. Journal of Cognition and Instruction, 13(2):161-188.

Ehud Reiter and Robert Dale. 2000. Building natural language generation systems. Cambridge University Press.

Verena Rieser, Oliver Lemon, and Xingkun Liu. 2010. Optimising information presentation for spoken dialogue systems. In *48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Somayajulu Sripada, Ehud Reiter, Jim Hunter, and Jin Yu. 2003. Generating english summaries of time series data using the gricean maxims. In *9th ACM international conference on Knowledge discovery and data mining (SIGKDD)*.

Somayajulu Sripada, Ehud Reiter, I Davy, and K Nilssen. 2004. Lessons from deploying NLG technology for marine weather forecast text generation. In *PAIS session of ECAI-2004:760-764*.

Richart Sutton and Andrew Barto. 1998. Reinforcement learning. MIT Press.

Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal Data Warehousing and Mining*, 3(3):1–13.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Random k-labelsets for multi-label classification. *IEEE Transactions on Knowledge and Data Engineering*, 99(1):1079–1089.

Grigorios Tsoumakas, Eleftherios Spyromitros-Xioufis, Josef Vilcek, and Ioannis Vlahavas. 2011. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12(1):2411–2414.

Marian van den Meulen, Robert Logie, Yvonne Freer, Cindy Sykes, Neil McIntosh, and Jim Hunter. 2010. When a graph is poorer than 100 words: A comparison of computerised natural language generation, human generated descriptions and graphical displays in neonatal intensive care. In *Applied Cognitive Psychology, 24: 77-89*.

Ian Witten and Eibe Frank. 2005. Data mining: Practical machine learning tools and techniques. Morgan Kaufmann Publishers.

Min-Ling Zhang and Zhi-Hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048.