Active Learning for Player Modeling

Noor Shaker
Center for Computer Games Research
IT University of Copenhagen
Rued Langgaards Vej 7
Copenhagen, Denmark
nosh@itu.dk

Mohamed Abou-Zleikha
Audio Analysis Lab, AD:MT
Aalborg University
Rendsburggade 14, Aalborg
Copenhagen, Denmark
moa@create.aau.dk

Mohammad Shaker
Joseph Fourier University
Grenoble, France
mohammadshakergtr@gmail.com

ABSTRACT

Learning models of player behavior has been the focus of several studies. This work is motivated by better understanding of player behavior, a knowledge that can ultimately be employed to provide player-adapted or personalized content. In this paper, we propose the use of active learning for player experience modeling. We use a dataset from hundreds of players playing *Infinite Mario Bros.* as a case study and we employ the random forest method to learn models of player experience through the active learning approach. The results obtained suggest that only part of the dataset (up to half the size of the full dataset) is necessary for the construction of accurate models that are as accurate as those constructed from the full dataset. This indicates the potential of the method and its benefits in cases when obtaining the data is expensive or time, storage or effort consuming. The results also indicate that the method can be used online during the content generation process where the models can improve and better content can be presented as the game is being played.

1. INTRODUCTION

Providing entertaining content is the ultimate goal of game design. Consequently, predicting player preference is considered to be one of the main tasks of AI in games and its importance is a clear-cut in several applications. Player modeling [43] is one way to capture how player behavior is related to her preference. Building such models however requires collecting representative and meaningful player information. Most of the approaches used for this task use machine learning techniques that learn from player behavior datasets annotated with player experience tags. While accurate models can be constructed using such methods, the process of designing the data collection surveys and gathering the needed information is time and effort consuming. Moreover, the stages on which the process flows are usually predefined and the data collection process usually continues up to a certain point when the analysts assume the dataset is rich enough to start conducting experiments. If, after running some analysis, this turns out not to be the case, data collection continues and the process is repeated.

One would argue that we are in the age of data where we have access to a vast volume of game logs. Most of the data available however is noisy, dirty, and it does not provide information about players' feedback. Labeling game logs is usually unfeasible and selecting part of the data to be manually labeled could be biased as it might not preserve the true distribution of the classes. Due to this limitation of manually or randomly selecting instances methods, an efficient method for unlabeled instance selection is required. If the method is able to identify instances to be labeled we could potentially speedup the learning process and reduce the amount of data required for labeling. Moreover, data collection and model construction could be run simultaneously ensuing enough data and accurate models before terminating the data gathering process.

From a game design point of view, the process of *intelligently* selecting the instances to learn from could be very beneficial within the game creation process; an initial average model of player from a relatively small set of data can be built and gradually improved by inquiring the player about feedback on certain, carefully chosen instances. This information can be seeded in the game creation process leading to the creation of better content.

The field of procedural content generation [29] has been relatively well explored and so is the field of player modeling [2]. Closing the loop between these two areas however is an important immature research direction with many exciting potentials [43]. This paper presents a step towards this goal by proposing a system that can be ultimately used for content personalization. We propose the use of active learning methods for the construction of accurate estimators of player experience driven by the in-game interaction. Active learning promises the possibility of learning accurate models from a small set of annotated data and effectively presenting content for labeling based on how much information is learned about a specific player. We present the method and we analyze its efficiency in terms of the data needed and the modeling accuracy. We finally present our vision of how the method can be employed during the content generation process.

2. PROCEDURAL CONTENT GENERATION AND PLAYER-DRIVEN PCG

Procedural content generation (PCG) has witnessed increasing attention in commercial games such as *Diablo* and *Spore*. The Search-based Procedural Content Generation [39] paradigm, which uses global stochastic search algorithms has been used for evolving different aspects of game content (e.g. levels [28, 35], weapons [8], tracks [37] and rulesets [38] as well as complete games [5]) for various game genres.

The generation of personalized content is becoming an important trend within PCG. Personalized tracks [36] and weapons [8] have been evolved relying on different variations of theories of fun. Empirical models of player experience that approximate the relationship between player behavior and game content have also been investigated [32, 27]. The ultimate aim of constructing the models is to permit a data-driven mechanism of adjusting the game to individual player taste [34]. While this research direction is interesting and could potentially improve player experience if applied in real-time, the current approaches to player modeling and game adaptation has so far suffered from major limitations: (1) data needs to be collected from a large number of players, the data should preferably include information about the game content, player behavior, player's feedback (gathered through subjective and/or objective measures) and perhaps demographics information. This process is usually time and effort consuming. (2) models of player experience are usually built offline and later used to estimate player experience. This means that the outcomes of the models are approximate estimations of the experience based on the training data. (3) This also mitigates the ability of generating personalized models as the models are fixed before shipment. (4) Optimizing aspects of player experience have usually been limited to a small predefined set of content aspects and (5) the selection of game configurations from which to gather the data has so far not been optimized for efficient model learning and is not influenced by player behavior of preferences.

3. ACTIVE LEARNING

Active Learning (AL) refers to a set of methods where the learning algorithm is allowed to select the data to learn from. The selection is based on the learning gain and therefore the main advantage of these methods is that they significantly reduce the amount of data needed for training [23]. AL methods are usually employed to build models in situation where data needs to be annotated and the annotation process is expensive (usually requires enquiring a human) or in situations where reducing the number of experiments to run is beneficial [15]. AL attempts to overcome the annotation bottleneck by effectively selecting a number a unlabeled instances to be labeled hence achieving high accuracy using as few labeled instances as possible. The methods are gaining rapid interest, as they allow one to build an optimal training set with a minimum of queries (or labeled instances). The method has successfully been implemented in a wide range of applications such as image classification, information extraction and speech recognition [40, 25, 42].

Active learning is well-motivated in the game domain (such as in many other machine learning fields) where player data may be abundant (as we are in the "age of data") but the data is noisy and labels are scarce or expensive to obtain. Yet the use of this method has so far been limited to very few, yet interesting, examples. Zook et al. [44] applied AL methods to tune the game parameters so that they best meet players' preferences. Their system aimed at providing feedback for game designers during the game design process and the investigations were limited to three content parameters. Normoyle et al. [15] employed AL to select the content configurations that help better design and track of player metrics. The approach however has not been implemented yet to model player experience, i.e. to construct models of players experience that effectively capture the relationship between the content of the game and the players' behavior.

4. THE BIG PICTURE: ACTIVE LEARNING AND ARTIFICIAL CURIOUSITY

Artificial curiousity (AC) has been discussed in the context of generating novel solutions in an unknown environment [20]. The main idea behind achieving a computational model of AC is the use of general reinforcement learning to "maximize not only external reward or achieving goals such as the satisfaction of hunger and thirst, but also intrinsic reward for learning a better world model, by creating/ discovering/ learning novel patterns in the growing history of actions and sensory inputs" [21].

Increasing the diversity of the individuals in a population or abandoning objectives in the search [9, 7] are two well-known methods implemented in the field of evolutionary computation with the purpose of generating novel solutions to a given problem. Algorithms from the field of active learning can be used to guide the learning process towards finding "interesting" patterns to learn from and reduce the processing cost by giving the algorithm the ability to choose the learning data [20]. In this sense, the algorithm can be regarded as being "curious" about its user. The idea is inspired by the exploration mechanisms employed by children driven by social learning such as emulating and imitation and intrinsic motivation [16] which drives curiousity in humans [3]. The framework has been implemented in several domains (e.g. generating curious agents [18], building control systems [18], and developmental robotics [19]). The definition of the framework makes it highly applicable for the creative generation of game content, or what we call *Playful Games* where the feedback coming from the player constitutes the external reward while intrinsic reward is given for exploring *surprise* content. This area of research has not been explored in the game domain yet.

5. THIS PAPER

We believe that better gameplay experience could be facilitated if games are allowed and equipped with techniques that permit exploration of the player as the player is exploring the game. More specifically, if the game is able to explore content configurations that it thinks are important to learn more about the player, i.e. to be curious about the player, it is most likely that over time the game will be able to deliver "better" content.

This paper constitutes the first step towards building a *Playful Game*. The model we propose uses active learning methods to maximize the intrinsic reward of learning a better model of players as more information about player behavior is being collected. This model can later be used by a curious agent together with an external rewarding mechanism (potentially using reinforcement learning methods [21]) to ultimately build a playful game.

In terms of the advantages our method promises over traditional player modeling and content adaptation approaches, AL methods are widely known for their ability to handle scarce data through efficiently choosing the "important" instances to learn from. In our context, AL permits a number of main advantages: (1) it facilitates the construction of accurate models while reducing the number of play testers required for data collection; (2) because AL works with small datasets, it is easer to be embedded in an online adaptation mechanism— in this scenario, the player experience models can be adjusted based on the data coming from a specific player and new content configurations are presented to the player that better match her preferences— and (3) the explored content space does not need to be limited or predefined, the method can search for the best area in this space to explore given the player response. This permits efficient content personalization, as exploration of content variations will be guided by the specific player's behavior and feedback.

In this paper we focus on point one while we leave points two and three for future work. In particular, we aim at utilizing AL methods for the construction of accurate estimators of players' experience. Similar to existing experiments where the goal of running additional experiments is to reduce the uncertainty in the data [15, 24], the goal of our setup is to select instances for the purpose of reducing the uncertainty in our player experience modeling.

As we are building models of player experience averaged across all players, we believe AL methods can be highly beneficial as they have the advantage of identifying rare instances and consequently ask for more information about those [15]. This means that we will ultimately be able to converge to accurate models faster with less training instances which also means that we are reducing the number of experiments to run while preserving the performance. To the best of our knowledge, we are the first to explore an active learning approach for player modeling and for content personalization.

We start by presenting how active learning works in Section 6. Section 7 discusses our methodology to implement AL methods for player experience modeling. For our analysis, we use a dataset collected from players playing *Infinite Mario Bros.* (SMB) as our testbed game as described in Section 8. We present and discuss the results obtained in Section 10. And finally, we conclude the paper and discuss future directions in Section 11.

6. ACTIVE LEARNING PROCEDURE

Building a model that learns from a given set of labeled data is what traditional supervised machine learning algorithms usually do. On the contrary, an active learner starts with a small training set of labeled instances L, and is given a degree of control that allows it to carefully choose instances from a large unlabeled pool U and ask an oracle Q for labeling of those. The newly labeled instance L_n are added to the training set and the learner can then learn from the resultant set and use this updated knowledge to select the instances to query next. After adding the newly selected instance to the training set, the learner proceeds in a standard supervised way. By following this process, the model is improved by focusing on the minimal set of labeled data that defines conflicting areas. Consequently, the models will be accurate and have generalizable capabilities [22, 25]. Algorithm 1 describes the general framework of the active learning process.

Algorithm 1 Active learning framework

```
$C$= build a classification/rating model from the labeled data $L$
$e$=classification error on the testing set
**while** ( $U$ is not empty and $e>threshold$) **do**
    $U_m$= get most $m$ uncertain instance from unlabeled data $U$ to be labelled using the classifier $C$
    $L_n$=label $U_m$ using an oracle $Q$
    $L$=$L \cup L_n$
    $C$= build a classification/rating model from the labeled data $L$
    $e$=classification error on the testing set
**end while**
```

In particular, a classifiers is built from the set of available labeled data L and the error is calculated on the testing set. The AL method then proceed by selecting a set of instances with high uncertainty and asking an oracle Q for labels for those. The newly labeled instances are then added to the dataset L and the process is repeated until the error becomes smaller than a predefined threshold or when there are no more unlabeled instances.

Several query methods have been proposed and they are usually referred to as *query strategies*. Several paradigms have been proposed for query strategies including uncertainty sampling, which relies on selecting the instances with the least certainty on how they should be labeled. Most of the proposed approaches in this category build probabilistic models [11, 25]. There exists, however, a number of non-probabilistic methods that also use the uncertainty sampling for instance selection such as the work done by Lewis et. al. [10] in which decision trees are used as classifiers and the work by Lindenbaum et. al [12] where K-nearest neighbor is employed. Demir et. al. [6] proposed the use of multiclass-level uncertainty via support vector machine as a classifier to calculate the confident values of the membership of each unlabeled instance to each class which is then used to calculate a distance function that is used to rate the instances.

The second paradigm is the Query-By-Committee (QBC). According to this approach, several models are constructed for the target problem. Unlabeled instances are annotated by each model and the instances with the most disagreement are considered the most informative. Active learning algorithms based on boosting and bagging have been proposed in several studies [13, 4, 41]. Melville et. al. [14] proposed the use of artificial data to increase the diversity between the classifiers, which as a result increases the disagreement between the query instances.

The Expected Model Change (EMC) and the expected error reduction paradigms are also used where the potential influence of the instances on the model or on the classification accuracy if they are labelled is considered an important factor [26, 17].

In this paper, a similar approach to the one proposed by Demir et al. [6] is used. According to this approach, the membership of each instance to all possible classes is calculated and rated. The difference of the memberships of the top two classes is then calculated and the instance with the lowest difference (the lower the difference the more uncertain the classification is) is the one chosen to be labelled. Demir et al. [6] combines this approach with SVM as a classification method. In our implementation however, instead of using the SVM to calculate the confident value of the membership of each unlabelled instance to each class, we used a random forest. Our decision is motivated by preliminary experiments that showed better performance for random forest over SVM on the chosen dataset.

7. ACTIVE LEARNING FOR PLAYER MODELLING

In the player modeling task, models are usually built offline from a predefined dataset collected about player behavior. This approach has a number of drawbacks as discussed previously and a better approach would be the online construction of the models which also facilitates building personalized models and providing better gameplay experience.

To implement AL for player modeling, we assume that each play through of the game is an instance associated with a label (the user's rating). Consequently, we suppose we have a set of labeled instances L and a set of unlabeled instances U. Following the steps of the AL algorithm described in 1, we start by building a classification model C. Then in each iteration, we select a number of samples from the unlabeled data to be labeled (rated) by players. This selection is done by calculating the probabilities of the membership of each unlabeled sample to each class (a class in this case is one of the possible values of the feedback provided by the user). Once the instances with the highest uncertainty are identified, the part of these instances that contain information about game content is parsed and used to generate new content instances, which are then presented to the player who plays and provide a feedback (label). The newly labeled instances are then added to the training dataset and a new classification model is built from the new set. The process is then repeated until we collect labels for all instances or until the classification error reaches a threshold.

8. DATA DESCRIPTION

The evaluation set in our experiments consists of player data for hundreds of players while playing a clone of the popular game Super Mario Bros. The clone is modified to permit control over level generation and thereafter provide different variations of content for players to experience and compare. The gameplay in Infinite Mario Bros consists of moving the player-controlled character, Mario, through two-dimensional levels. Mario can walk and run, duck, jump, and shoot fireballs. The main goal of each level is to get to the end of the level. Auxiliary goals include collecting as many coins as possible, and clearing the level as fast as possible. The player is given three lives to complete the game.

An experiment is conducted to collect data from players playing the game. According to the original protocol followed, players are presented with a pair of two sessions that differ along one or more aspects of game content. After completing each level, a Likert questionnaire scheme is presented and the player is asked to express her emotional preferences across the three different emotional states (engagement, frustration and challenge). The likert scales from 0 to 4 representing the strength of the emotion (4 means "extremely"; 0 means "not at all"). And after playing each pair, the players were asked to report their preferred game for the same emotional dimensions following the four-alternative forced choice protocol. This setting was followed to allow studies of different user modelling methods from rating and pairwise preferences. In this paper, we focus only on players' responses provided as rating.

An executable version of the software was uploaded online and participants were invited to play the game and answer the questionnaire. The data was collected over a period of six months where 273 unique players participated. Participants' age covers a range between 16 and 64 years (31.5% females).

Several representative features of player behavior were extracted and Table 1 presents a subset of these features. The full set contains 30 features which can be found in [33]. Each game log consists of the set of these features along with another set of content parameters that capture different aspects of the game content presented to the players.

The dataset has been extensively used in previous research for modeling player experience [30, 31, 33] and as a testbed for preference learning [1]. In this paper, we focus on modeling players' reported experience of frustration and challenge as they are very important factors in game design and as we believe the results obtain can be easily scaled to construct models for predicting reported engagement (empirical evaluation however constitutes a future work).

9. EXPERIMENTAL SETUP

The complete datasets used consist of 1062 and 1258 gameplay sessions for frustration and challenge, respectively. For model construction, the dataset was split into 80% for training and 20% for testing. Notice that in this particular case and in order to apply AL approach on the collected dataset, we assume that labels are available for only part of the data and querying players actually means retrieving the rates for the chosen instances from the unseen data. Notice that this does not limit nor affect the performance of the approach as the method will behave exactly the same if it is to collect feedback online. The initial training labelled instances have been randomly selected. As previously mentioned, the random forest classifier has been used to calculate the confident value of the membership of each unlabelled instance to each class and to calculate the rating error as well. The error function used is defined as:

$$e(x,y) = \begin{cases} 0 & x = y \\ 1 & \text{otherwise} \end{cases} \tag{1}$$

where x is the predicted rating and y is the actual rating.

10. ACTIVE PLAYER MODELING

For our experiments, we use the SMB dataset. The goal is to start from a smaller set of data where labels (rating in our case) are available and to investigate whether the proposed AL method can be used to effectively construct accurate models. For this purpose, we conduct a number of experiments to investigate the general behavior of the method and the best configuration for efficient modeling.

10.1 The Size of the Initial Labeled Training Set

One of the crucial issues in active learning is the size of labeled data, L, initially used for training. The optimal size of this set affect the accuracy and the time required for the method to converge. This size also infers the experiment design process as it defines the minimal set size for constructing accurate models.

To investigate this issue, we conduct a study to analyze the effect of the initial labeled data size on the accuracy of the active learning process; we run a number of experiments where the initial labeled data size ranges from 10%, 20%, 40% to 70% of whole set. The AL can ask for 10 samples to be labeled in each learning iteration. Fig. 1 presents the average modeling accuracy obtained from 50 runs of this experiment when applied on the frustration and challenge datasets. The results indicate that all models eventually converge to the same small error rate. The size of the initial training set L, however has a great impact on the amount of data to be labeled. Models of very small initial sizes for L (10% and 20%) ask for more data and converge only when the size of L becomes larger than 50% of the full data size. Models that initially use 70% of the data, on the other hand, converge significantly faster (according to t-test) and ask for considerably less amount of data for labeling.

As expected, there is clearly a tradeoff between the size of L, the modeling accuracy and the modeling time. The smaller the size the more data inquired and the slower the modeling process. At some point in this process however, the significance of adding more labeled instances becomes smaller than the expense of labeling it and this point defines the optimal size of the training dataset. It is also interesting to note that in the first three cases (using 10%, 20% or 40% as initial data) models that are as accurate as those constructed using 70% of the data were built when the size of the set becomes only about 50% of the full set. This suggests that one

Table 1: Subset of representative features extracted from Infinite Mario Bros dataset.

Category	Feature	Description
Time	t_{comp}	Completion time
	t_{play}	Playing duration of last life over total time spent on the level
	t_{jump}	Time spent jumping (%)
	t_{left}	Time spent moving left (%)
	t_{right}	Time spent moving right (%)
	t_{run}	Time spent running (%)
	t_{small}	Time spent in Small Mario mode (%)
	t_{big}	Time spent in Big Mario mode (%)
Interaction with items	n_{coin}	Free coins collected (%)
	$n_{coinBlock}$	Coin blocks pressed or coin rocks destroyed (%)
Interaction with enemies	k_{goomba}	Times the player kills a goomba or a koopa (%)
	k_{stomp}	Opponents died from stomping (%)
Death	d_{total}	Total number of deaths
	d_{cause}	Cause of the last death
Miscellaneous	n_{mode}	Number of times the player shifted the mode (Small, Big, Fire)
	n_{jump}	Number of times the jump button was pressed

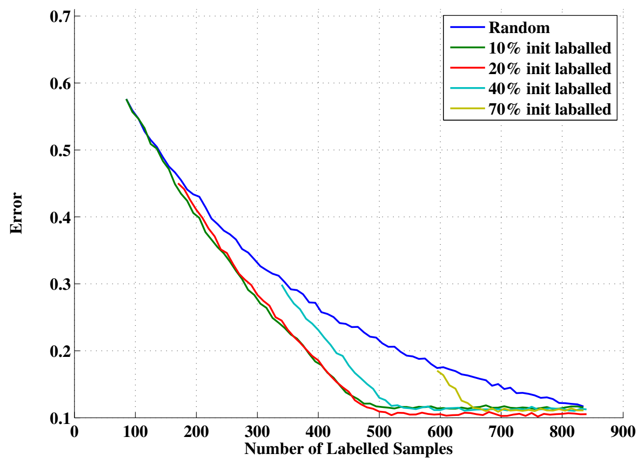

Figure 1: The average accuracies for predicting *frustration* obtained from 50 runs for each configuration using the whole training data (850 samples). Each configuration represents the amount of the initial labeled data used to build the model. In each iteration the 10 most uncertain samples are selected and labeled.

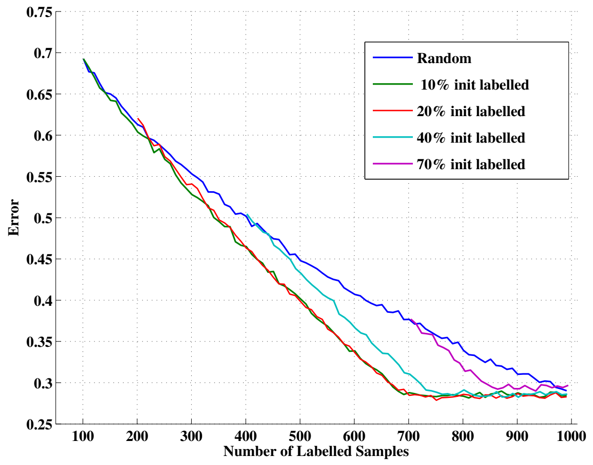

Figure 2: The average accuracies for predicting *challenge* obtained from 50 runs for each configuration using the whole training data (1006 samples). Each configuration represents the amount of the initial labeled data used to build the model. In each iteration the 10 most uncertain samples are selected and labelled.

would need only half the size of the training set to build as accurate models as those built from the whole data.

In order to check whether this is a general trend or a specific issue in the tested dataset, the same experiment is repeated for the construction of models for predicting challenge and the results are presented in Fig. 2. The results obtained support the claims presented previously and emphasize the importance of the initial size of L.

It is interesting to note however that the results indicate that challenge is harder to predict than frustration as the models achieved higher error rate and required more data to learn from.

An interesting observation is that the AL converges to the smallest error using less than the full data available in all cases. In practice, this means that we could terminate the process of data collection as soon as we detect a stabilization in the behavior of the system.

10.2 The Size of the Learning Batch

During the learning process and in each iteration, the learner can choose a number of instances to be labeled (called the learning batch). The size of this set impacts the speed of the modeling process. To empirically investigate this effect, we run a number of experiments where the size of the learning batch is set to 1, 10, 20 or 50 samples. The experiment is conducted on the frustration and challenge datasets. A classifier with a random selection of samples that uses one sample per truing iteration is also presented for comparison. The results obtained presented in Fig. 3 and Fig. 4 illustrate that models that uses batches of smaller sizes are more efficient than the others. In particular, the models that uses a batch of size one yield the smallest size of training data while achieving similar performance to the other models. The results are not very obvious as one would expect that larger batch size would give the model more knowledge faster and thereafter the model is expected to converge with less data. The behavior obtained can be explained by the process of learning. When a small batch size is used, the AL fully grasps the new knowledge and searches for the next best

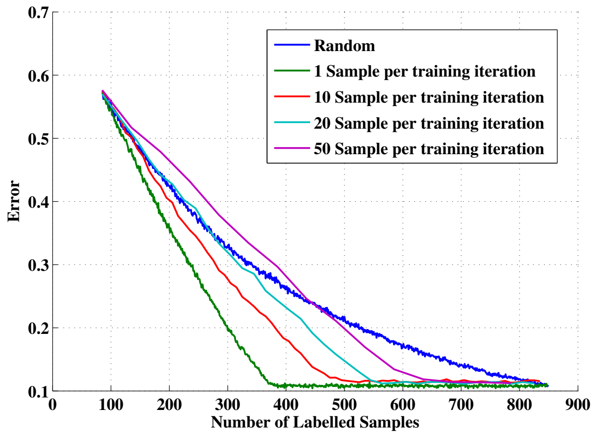

Figure 3: Average accuracies for predicting *frustration* obtained from 50 runs for each configuration using the full dataset (850 samples). Each configuration differs by the size of the learning batch (samples to be labelled in each iteration).

sample to inquiry. On the contrary, the information gain of batch of size 20 for instance is smaller as the learner chooses the 20 samples to be labeled in one iteration and misses the chance to learn from each sample individually.

10.3 Modelling Accuracies

The models constructed in our previous experiment suggest that models of high accuracies can be built using the AL approach. The models built using the full dataset and a learning batch of size 10 for instance, achieved very accurate results (88.4% and 71.4% for predicting frustration and challenge, respectively) that are comparable to the best presented in the literature [33] using only part of the dataset. The results indicate the efficiency of the proposed approach in modeling player experience and in exploring the search space.

10.4 Scalability

In previous sections, we described how the method works and illustrated the results in cases where the full dataset was already available. The results indicate that by using smaller portion of the data, models that are as accurate as those constructed from the full dataset can be constructed. It is interesting however to examine the method behavior on datasets from smaller sizes and to check whether this claim still holds. For this purpose, we randomly chose 425 and 503 instances from each of the dataset and treated these new subsets as our full datasets. The experiment presented in Section 10.1 is then repeated and the results obtained for predicting frustration and challenge are presented in Fig. 5 and Fig. 6, respectively.

As expected, the models constructed are slightly less accurate than those constructed from the full set but the results indicate a similar behavior in general. The different configurations in both cases gradually improve the performance as they inquiry more instances. In the latter case (using half the size of the data) the models converge faster with fewer amounts of labeled data. This of course comes with the price of the higher error rate. The results suggest however that the relationship between the size of the full dataset and the amount of data that needs to be labeled is a non-linear one.

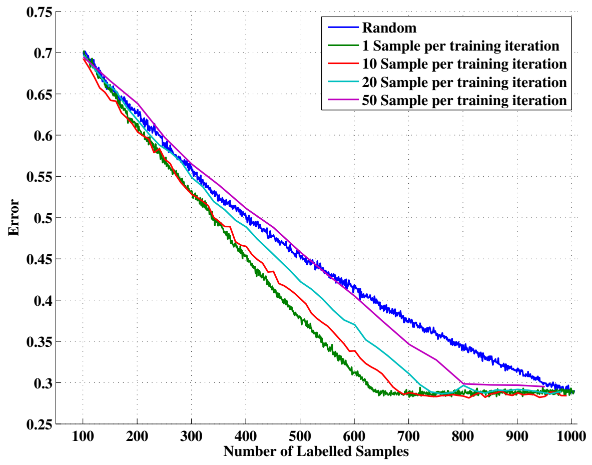

Figure 4: Average accuracies for predicting *challenge* obtained from 50 runs for each configuration using the full dataset (1006 samples). Each configuration differs by the size of the learning batch (samples to be labelled in each iteration).

This also relates to the amount of information and its distribution in the dataset. In our latter case for predicting frustration, the models seems to learn almost all they need from about 70% of the data and thereafter more information would be redundant while In the first case (Fig. 1), about 60% of the data were required to stabilize the accuracy. This demonstrates the efficiency of the method as it achieved high accuracy by effectively exploring the search space.

11. CONCLUSIONS

This paper presents the use of active learning method for modeling player experience. Active learning is an efficient approach presented in the literature to effectively learn from as less data as possible. This is achieved by intelligently exploring the search space and inquiry only about informative instances. The approach speeds up the learning process and largely reduces the amount of data needed for training. We present how active learning generally works and we describe its instantiation for our player modeling task. We use a dataset of player behavior while playing Infinite Mario Bros. as a test case and we build models for predicting reported frustration and challenge. We run several experiments to investigate the performance, the modeling accuracy and the scalability of the method. The results show that models of high accuracies can be built using smaller portions of the original dataset. The results also suggest that the distribution of the training data has a great impact on the amount of data needed to construct accurate models.

This work present the first step in building a full framework for content personalization. The models constructed are to be used online during the content generation process as they permit effective selection of content instances for the player to explore. Player's behavior is then captured and her feedback is collected as needed. As the process continues, personalized models are constructed that effectively explore the player as she explores the game ultimately leading to a better gameplay experience.

Another interesting future direction is the use of the proposed method to rank instances of game content. This requires the construction of models trained only on content parameters so that they can be used

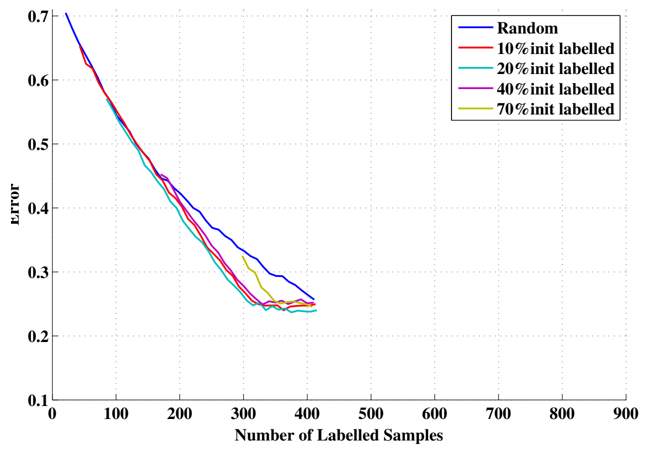

Figure 5: Average accuracies for predicting *frustration* obtained from 50 runs of each configuration using half the size of the training data (425 samples). In each iteration, 10 samples are chosen to be labeled by the oracle.

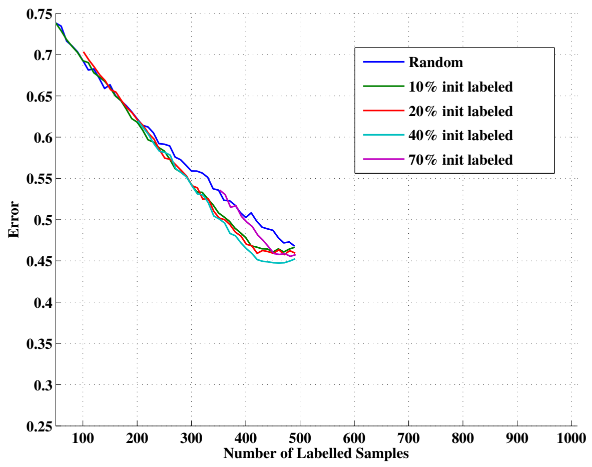

Figure 6: Average accuracies for predicting *challenge* obtained from 50 runs of each configuration using half the size of the training data (503 samples). In each iteration, 10 samples are chosen to be labeled by the oracle.

to predict the appeal of a piece of content before the actual play through. This could be beneficial for game designers who would like to get average feedback on their design before shipping the game.

12. ACKNOWLEDGEMENTS

The research was supported in part by the Danish Research Agency, Ministry of Science, Technology and Innovation; project "Play-GALe" (1337-00172). This work also was supported in part by the Danish Council for Strategic Research of the Danish Agency for Science Technology and Innovation under the CoSound project, case number 11-115328. This publication only reflects the authors views.

13. REFERENCES

[1] M. Abou-Zleikha and N. Shaker. Evolving random forest for preference learning. In *Applications of Evolutionary Computation*, pages 318–330. Springer International Publishing, 2015.

[2] S. C. Bakkes, P. H. Spronck, and G. van Lankveld. Player behavioural modelling for video games. *Entertainment Computing*, 3(3):71–79, 2012.

[3] A. Baranes and P.-Y. Oudeyer. Robust intrinsically motivated exploration and active learning. In *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*, pages 1–6. IEEE, 2009.

[4] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 49–56. ACM, 2009.

[5] C. Browne and F. Maire. Evolutionary game design. *IEEE Transactions on Computational Intelligence and AI in Games,*, 2(1):1–16, 2010.

[6] B. Demir, C. Persello, and L. Bruzzone. Batch-mode active-learning methods for the interactive classification of remote sensing images. *Geoscience and Remote Sensing, IEEE Transactions on*, 49(3):1014–1031, 2011.

[7] F. J. Gomez, J. Togelius, and J. Schmidhuber. Measuring and optimizing behavioral complexity for evolutionary reinforcement learning. In *Artificial Neural Networks–ICANN 2009*, pages 765–774. Springer, 2009.

[8] E. J. Hastings, R. K. Guha, and K. O. Stanley. Evolving content in the galactic arms race video game. In *Proceedings of the 5th international conference on Computational Intelligence and Games*, CIG'09, pages 241–248, Piscataway, NJ, USA, 2009. IEEE Press.

[9] J. Lehman and K. O. Stanley. Exploiting open-endedness to solve problems through the search for novelty. *Artificial Life*, 11:329, 2008.

[10] D. D. Lewis and J. Catlett. Heterogenous uncertainty sampling for supervised learning. In *ICML*, volume 94, pages 148–156, 1994.

[11] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.

[12] M. Lindenbaum, S. Markovitch, and D. Rusakov. Selective sampling for nearest neighbor classifiers. *Machine learning*, 54(2):125–152, 2004.

[13] N. A. H. Mamitsuka. Query learning strategies using boosting and bagging. In *Machine Learning: Proceedings of the Fifteenth International Conference (ICML'98)*, page 1. Morgan Kaufmann Pub, 1998.

[14] P. Melville and R. J. Mooney. Diverse ensembles for active learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 74. ACM, 2004.

[15] A. Normoyle, J. Drake, M. Likhachev, and A. Safonova. Game-based data capture for player metrics. In *AIIDE*, 2012.

[16] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *Evolutionary Computation, IEEE Transactions on*, 11(2):265–286, 2007.

[17] T. Scheffer, C. Decomain, and S. Wrobel. Active hidden markov models for information extraction. In *Advances in*

Intelligent Data Analysis, pages 309–318. Springer, 2001.

[18] J. Schmidhuber. Curious model-building control systems. In *Neural Networks, 1991. 1991 IEEE International Joint Conference on*, pages 1458–1463. IEEE, 1991.

[19] J. Schmidhuber. Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, 18(2):173–187, 2006.

[20] J. Schmidhuber. Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. In *Anticipatory Behavior in Adaptive Learning Systems*, pages 48–76. Springer, 2009.

[21] J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *Autonomous Mental Development, IEEE Transactions on*, 2(3):230–247, 2010.

[22] B. Settles. *Curious machines: active learning with structured instances*. ProQuest, 2008.

[23] B. Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52:55–66, 2010.

[24] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[25] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079. Association for Computational Linguistics, 2008.

[26] B. Settles, M. Craven, and L. Friedland. Active learning with real annotation costs. In *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, 2008.

[27] N. Shaker, S. Asteriadis, G. N. Yannakakis, and K. Karpouzis. Fusing visual and behavioral cues for modeling user experience in games. *Cybernetics, IEEE Transactions on*, 43(6):1519–1531, 2013.

[28] N. Shaker, M. Nicolau, G. N. Yannakakis, J. Togelius, and M. O'Neill. Evolving levels for super mario bros using grammatical evolution. In *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*, pages 304–311. IEEE, 2012.

[29] N. Shaker, J. Togelius, and M. J. Nelson. *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*. Springer, 2015.

[30] N. Shaker, G. Yannakakis, and J. Togelius. Digging deeper into platform game level design: session size and sequential features. *Applications of Evolutionary Computation*, pages 275–284, 2012.

[31] N. Shaker, G. N. Yannakakis, and J. Togelius. Feature Analysis for Modeling Game Content Quality. In *IEEE Conference on Computational Intelligence and AI in Games (CIG)*, pages 1–16, 2010.

[32] N. Shaker, G. N. Yannakakis, and J. Togelius. Crowd-sourcing the aesthetics of platform games. *IEEE Transactions on Computational Intelligence and AI in Games*, 2012.

[33] N. Shaker, G. N. Yannakakis, and J. Togelius. Crowdsourcing the aesthetics of platform games. *Computational Intelligence and AI in Games, IEEE Transactions on*, 5(3):276–290, 2013.

[34] N. Shaker, G. N. Yannakakis, J. Togelius, M. Nicolau, and M. O'Neill. Evolving personalized content for super mario bros using grammatical evolution. In *AIIDE*, 2012.

[35] N. Sorenson, P. Pasquier, and S. DiPaola. A generic approach to challenge modeling for the procedural creation of video game levels. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3):229–244, 2011.

[36] J. Togelius, R. De Nardi, and S. Lucas. Towards automatic personalised content creation for racing games. In *IEEE Symposium on Computational Intelligence and Games, 2007. CIG 2007*, pages 252–259. IEEE, 2007.

[37] J. Togelius, R. D. Nardi, and S. M. Lucas. Making racing fun through player modeling and track evolution. In *Proceedings of the SAB'06 Workshop on Adaptive Approaches for Optimizing Player Satisfaction in Computer and Physical Games*, 2006.

[38] J. Togelius and J. Schmidhuber. An experiment in automatic game design. In *IEEE Symposium On Computational Intelligence and Games. CIG'08*, pages 111–118. IEEE, 2008.

[39] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne. Search-based procedural content generation. In *Proceedings of EvoApplications*, volume 6024. Springer LNCS, 2010.

[40] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 107–118. ACM, 2001.

[41] D. Tuia, F. Ratle, F. Pacifici, M. F. Kanevski, and W. J. Emery. Active learning methods for remote sensing image classification. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(7):2218–2232, 2009.

[42] G. Tur, D. Hakkani-Tür, and R. E. Schapire. Combining active and semi-supervised learning for spoken language understanding. *Speech Communication*, 45(2):171–186, 2005.

[43] G. N. Yannakakis and J. Togelius. Experience-Driven Procedural Content Generation. *IEEE Transactions on Affective Computing*, 2011.

[44] A. Zook, E. Fruchter, and M. O. Riedl. Automatic playtesting for game parameter tuning via active learning. In *Proceedings of the 9th International Conference on the Foundations of Digital*, 2014.