Learning Multimodal Entity Representations and Their Ensembles, with Applications in a Data-Driven Advisory Framework for Video Game Players[☆,☆☆]

Andrzej Janusz[a,b], Daniel Kałuża[a,b], Maciej Matraszek[a,b], Łukasz Grad[a], Maciej Świechowski[b], Dominik Ślęzak[a,b]

[a] *Institute of Informatics, University of Warsaw, Warsaw, Poland*
[b] *QED Software, Warsaw, Poland*

Abstract

We investigate the impact of combining multiple representations derived from heterogeneous data sources on the performance of machine learning (ML) models. In particular, we experimentally compare the approach in which independent models are trained on data representations from different sources with the one in which a single model is trained on joined data representations. As a case study, we discuss various entity representation learning methods and their applications in our data-driven advisory framework for video game players, called SENSEI. We show how to use the discussed methods to learn representations of cards and decks for two popular collectible card games (CCGs), namely Clash Royale (CR) and Hearthstone: Heroes of Warcraft (HS). Then, we follow our approach to create ML models which constitute the back-end for several out of SENSEI's end-user functionalities. When learning representations, we consider techniques inspired by the NLP domain, as they allow us to create embeddings which capture various aspects of similarity between entities. We put them together with representations composed of manually engineered features and standard bags-of-cards. On top of that, we propose a new *end2end* deep learning architecture with an attention mechanism aimed at reflecting meaningful inter-entity interactions.

Keywords: Video Game Analytics, Representation Learning, Multimodal Data Analysis, Ensembles in Machine Learning, Attention in Neural Networks

[☆] The formal publication of this Accepted Manuscript is available at https://doi.org/10.1016/j.ins.2022.10.097

Email addresses: janusza@mimuw.edu.pl (Andrzej Janusz), daniel.kaluza@qed.pl (Daniel Kałuża), m.matraszek@mimuw.edu.pl (Maciej Matraszek), lgrad@uw.edu.pl (Łukasz Grad), maciej.swiechowski@qed.pl (Maciej Świechowski), slezak@mimuw.edu.pl (Dominik Ślęzak)

1. Introduction

The analytics of data from computer games is getting increasingly growing attention from the data science community. Among many reasons for this phenomenon is the fact that video game players constitute a large share of the population. According to reports by `wepc.com`[1] and Grand View Research[2], there are more than 2.5 billion players all around the world. Only in the U.S., 68% of households host at least one person who plays video games at least 3 hours per week. This number would be even greater if we added people who are interested in eSports and follow news related to the growing number of international eSport events. There is also a large group of fans who search the web and dedicated social media (e.g., Discord, Twitch) for information regarding games they like, even though they do not play by themselves. Intelligent game data analytics aims at answering the needs of those heterogeneous social groups.

One of the pivotal issues related to video game data analytics is the fact that games, especially mobile video games, generate enormous amounts of data. This data may come from various sources, such as game logs, textual descriptions, databases of game results, and other meta-data. The utilization of such data in practice requires a proper identification of in-game entities, concepts, and complex relations between them. To this end, we need to develop a representation of the analyzed concepts, which is appropriate for the given task. The question is whether can we take advantage of the richness of available data sources and come up with a unified method for constructing such representations.

The considered problem of putting together multiple data representations computed using multimodal data sources is one of fundamental issues in SENSEI – the software framework which we developed in order to provide support and advice to video game players [19]. Accordingly, we discuss several entity representation learning mechanisms that were deployed in the first proof-of-concept of SENSEI that is focused on collectible card games (CCGs). In particular, we adapt methods inspired by techniques from the NLP domain for the purpose of learning representations of game cards using various data sources. We also propose a new *end2end* approach for learning representations of cards and decks, that uses the multihead attention mechanism to discover meaningful card interactions. We demonstrate the effectiveness of our approach on data from two popular mobile video games – *Clash Royale* (CR) and *Hearthstone: Heroes of Warcraft* (HS). Both of those games were the topic of research described in the literature [13] and can be seen as good representatives of the CCG genre.

This paper extends our previous research in the above areas [18]. Not only do we show that the constructed representations capture similarities between cards, but we also confirm their usefulness for ML problems related to the video game analytics, such as the estimation of deck win-rates and active learning of deck archetypes. We also illustrate how our aforementioned *end2end* attention-based

[1] `https://www.wepc.com/news/video-game-statistics/`
[2] `https://www.grandviewresearch.com/industry-analysis/video-game-market`

model, in addition to learning useful card representations, can be employed to discover effective counter-plays to individual game cards. Finally, we investigate the impact of combining multiple heterogeneous representations on the performance of ML models. We compare the approach in which independent models are trained on data representations from different sources with the one in which a single model is trained on joined data representations. One of our most interesting observations is that often these two approaches are worth combining, i.e., it is beneficial to create an ensemble in which some models are built on individual embedding-based representations while the other ones are learned on a richer, "concatenated" representation of all data sources.

Even though the application discussed in this paper is related to a single game genre, we believe that the methods which we describe have much wider use. They may be easily transferred to any game that involves using collections of entities, e.g., custom compositions of equipment, characters in a team, troops in an army, etc. Moreover, the real-life applications of the considered techniques may go beyond the domain of computer games. For instance, an analogous approach could be taken to learn representations of soccer players and teams to facilitate a prediction of game outcomes. Even broader, one can consider the application areas which are not related to any games at all. For example, the process-mining-related methods aimed at planning and optimizing various kinds of actions and procedures (in industry and logistics [1], hospital information systems [42], etc.) often need to learn from diverse information sources.

The remaining of the paper is organized as follows: In Section 2, we review the literature and explain relations between our current research and other domains. In Section 3, we describe SENSEI and its commercial deployments up to now. Apart from giving its overview, we present examples of its functionalities that are based on intelligent data analysis and require proper representations of game cards and decks. In Section 4, we discuss various techniques of learning embeddings of cards based on various data sources, such as databases of game results, collections of decks formed by players, textual descriptions of cards and their mechanics, and their basic characteristics extracted from game data. We introduce a novel architecture of a deep neural network (DNN) which learns interactions between cards to, e.g., detect efficient counter-plays (Subsection 4.4). This architecture provides us with one more type of embedding that can be used for building ensembles of ML models based on diverse data representations (Subsection 4.5). In Section 5, we report conducted experiments. In the tests, we check how different representations influence the quality of prediction models in tasks related to the functionalities of SENSEI. In Section 6, we analyze the obtained results. We pay particular attention to the investigated scenarios of combining multimodal embeddings. Section 7 concludes the paper.

2. Related Work

The video games area is a popular test-bed in AI studies. This stems from the fact that solutions to many game-related problems can be easily transferred to real-life issues, such as planning, real-time decision making and general AI [24].

Many researchers focus on development of bots capable of playing at human or even super-human level. Examples include applications in complex video games such as Doom [27] or Starcraft [23]. This refers to the domain of ML as well. For instance, for the aforementioned Starcraft, it is reported in [21] how to use generative adversarial neural networks to decode partially observed states from accumulated feature maps (images). As another example, in [5] one can read about the first AI system which defeats the world champions at an eSports game. The system builds an inner gradient-learned representation of the game state. Actually, it is constructed hierarchically using embedding, dense, convolutional and aggregation layers to combine simple entities and hand-crafted features into representations of units and teams. However, it requires an enormous amount of data and processing power, which is prohibitive for a typical application.

Although the above works constitute a great source of inspiration how to learn game representations, they are aimed at creating powerful artificial players. On the other hand, in this paper we are interested in using ML for video game data analytics [12]. Such analytics can be useful at many stages, starting from the process of game development and finishing with advisory tools for game players. We go back to this topic in Section 3, where we introduce the functionality and ML-related aspects of our SENSEI framework [19]. Let us mention that analogous solutions can be considered also for other domains, particularly when it comes to integrating domain knowledge into ML models [6].

Video game analytics was also a topic of several data mining competitions. One of them was related to the most recent deployment of SENSEI, namely a video game called *Tactical Troops: Anthracite Shift* (TT)[3]. In this case, the input data was available in a form of gameplay logs, aggregated data summaries, and screenshots. Such a rich, multimodal data greatly facilitated training ML models for predicting game winners [34][4]. We also organized two ML competitions related to CCGs and the problem of predicting deck win-rates: *AAIA'18 Data Mining Challenge: Predicting Win-rates of Hearthstone Decks* [20][5] and *Clash Royale Challenge: How to Select Training Decks for Win-rate Prediction* [17][6]. Data sets provided for those competitions are still openly available.

Different studies show that a proper input representation is often the key to the ML model performance [4, 41]. In this context, feature engineering was a topic of extensive research [25]. It has been demonstrated that constructing additional features often improves ML models' generalization capacity. However, practical studies show that when too many new features are added, the performance of classifiers often deteriorates due to the so-called, curse of dimensionality [3]. Various methods of dealing with this phenomenon were proposed, ranging from unsupervised dimensionality reduction, e.g., LSA [43], to supervised feature selection [29]. More recently, a novel technique of automatic

[3]`https://tacticaltroops.net/`
[4]`https://knowledgepit.ml/predicting-victories-in-video-games/`
[5]`https://knowledgepit.ml/predicting-winrates-of-hearthstone-decks/`
[6]`https://knowledgepit.ml/clash-royale-challenge/`

feature engineering based on reinforcement learning was proposed [26]. In other work [37], authors propose a technique for learning automatic data transformations that can be applied on new data without relying on model evaluation.

Various representations are suitable for different tasks. When it comes to CCGs, it is worth beginning with the methods of Natural Language Processing (NLP), whereby representation learning is very common. The state of the art results are often achieved through unsupervised model pretraining. Then, knowledge can be extracted from the model in form of word embeddings or the model can be further fine-tuned to a specific NLP task. Older approaches used to learn word representations such as *word2vec* [36] (e.g., CBOW or skip-gram model) from word context in a big text corpora. Recent language models such as BERT [8] are capable of distilling information connected to the specific word from the whole sentence and incorporate it into the contextualized word embedding. Therefore, they can obtain distinguishable representations of homonyms and capture the meaning of words in a more interpretable way.

Contextualized representation learning is popular also outside of the field of NLP, mainly due to the common incorporation of the attention-based mechanism into various DNN architectures. The attention layers allow DNNs to learn which parts of the input data are more important than others depending on the given context. In computer vision, the attention mechanism can be used to learn a contextualized representation of image patches [10]. In [2], the attention is used to sequentially select features important for the prediction performance on tabular data sets. The authors show that this approach increases model quality and makes it more interpretable at the same time. The attention-based approach can be also used to learn set representations that are permutation-invariant [28]. This task is related to the problem of learning deck embeddings in Subsection 4.4. Moreover, the problem of co-attention involves attending jointly to multiple data sources [31]. The architecture proposed in Subsection 4.4, is an example of co-attention to both decks simultaneously. The attention mechanism can be utilized also to integrate multimodal data, e.g., in the task of learning so-called social image embeddings. In [16], the authors propose a deep attention model to jointly embed multimodal input composed of image, text, and link data. This approach can be compared to our own way of working with multimodal data, although in our case, we use the attention outcomes only as one of many embeddings, rather than the integrator of all modalities.

In case of traditional ML classifiers, knowledge from various sources can be incorporated in the representation of input data or used during feature or hyper-parameter selection [11]. At the same time, ensembles usually outperform single model predictors in most of cases [9]. The topic of constructing reliable, yet not overly complex model ensembles was widely studied in the context of the diversification of the data representation [15]. In particular, combining models trained on randomized feature subsets is a commonly used technique [22]. An appropriate model selection and voting method can boost the predictive performance of a model ensemble [48], especially in case of imbalanced multi-label classification problems [14]. Ensemble learning methods were also applied for constructing similarity measures appropriate for various classification tasks [47, 49]. Therefore, we explore ensemble approaches which can achieve good results on various tasks, and may be used as a baseline out-of-the-box solution that supports also the future, yet not thoroughly specified analytical functionalities.

The main motivation for this paper was to explore and compare methods of constructing ensembles of diverse multimodal data representations. Let us refer to our own already-mentioned research on learning representations of cards and decks in CCGs [18]. Although useful as a background, that research did not embrace the crucial topic of putting different representations together into ensembles yet. Our new way of building multimodal ensembles can be compared with a number of other approaches to heterogeneous data processing, including infrared images, sketches, depth images or text descriptions [30, 46]. Herein, one can compare late integration methods that combine decisions based on each modality by voting or scoring and, on the other hand, early integration methods that merge features from different modalities by concatenation or learning a joint representation [39]. Interestingly, the results reported in Section 5 show that these two different strategies can be successfully blended together.

3. Analytics of CCG Data in SENSEI

For every eSports game, there are dedicated web portals, such as MobAnalytics[7] or HsReplay[8], providing game data and statistics that players can use to gain insights about various aspects of the game. Players can visit such sites, e.g., to learn popular starting strategies, get familiar with the current trends in meta-game (i.e., the most popular and efficient play strategies at the given time) or simply check preferences of others. The one thing that is missing in the most portals are mentoring tools for less experienced players. Often, only advanced players are able to take advantage of information provided by portals thanks to their experience. The rest of the community has to rely on generic internet resources or ask for recommendations from fellow players on social media. However, recommendations like that are rarely customized for particular players' situation, and thus, may be inconsistent with their playing style.

The SENSEI framework was designed to cope with the above issues. It provides in-depth data analysis and advice for players. Its main goal is to support players in the continuous development of their skills by giving them personalized feedback about their performance and providing ML-based tools and game-related recommendations. For the first proof-of-concept of SENSEI, we chose the aforementioned CCG game Clash Royale (CR). CCG is a game genre in which in-game progress of players is related to the acquisition of new cards. Players start with a small set of cards and gather new ones through gameplay or in-app purchases. This type of games usually involves some sort of players' ranking indicating their skills in comparison to other competitors.

[7]`https://mobalytics.gg/`
[8]`https://hsreplay.net/`

Players form a strongly motivated community, they want to improve their skills, extend their cards collection, and climb the game ranking.

3.1. Data-driven game advisor

For many online video games, the meta-game is dynamically evolving. Therefore, the models used to solve analytical problems, such as prediction of deck win-rates for CCGs, should be always fitted with the newest data. To achieve the best accuracy of these models, SENSEI needs to collect as much data as possible, therefore a high performance and scalable data acquisition architecture is proposed [19]. The architecture is capable of efficiently processing and storing a stream game logs. The potential volume of data in a month can be estimated by $N_{players} \cdot S$, where $N_{players}$ is an approximated number of active players in the month and S is an average size of one game log. CR was estimated to have 22 000 000 active players in March 2019[9] and the average size of one game log is 23 kB[10], which results in more than 506 GB of data generated by the players during a month. HS was estimated to have 5 000 000 players in March 2019[11], however, as HS does not have an official API with game logs available, there is also no reasonable lower bound estimation on the log data size. Nevertheless, available data grows by hundreds of GBs per month. Therefore, the collection of data is modularized in a way that every resource fetching can be independently horizontally scaled. All the retrieved data is then deposited in Apache Kafka[12] for further processing and backing up in MongoDB or HDFS. After retrieval from Kafka, processing of data takes place. This step can also be scaled up in the same manner. The procedure ends with transformed data mapped to the data schema and inserted to a relational database. Stored data can be retrieved by an application and presented to the user or passed to ML algorithms.

As visible in Figure 1, SENSEI takes advantage of all data sources that are available for a particular game. Data sources that can be obtained depend strongly on many factors like: the game type, the game developers studio approach to data sharing, and the game community. In practice, information about the game can be obtained among the others from: players in-game history, game logs, constants representing in-game entities or even articles about game mechanics. Diversity between these data sources requires methods of unification and representation in a way which enables further use in ML models or interpretation by an intelligent analytical system. Because the best results are obtained when the embedding method is matched to the data type, we explored various representation methods, therefore utilizing all available data.

3.2. Reaching beyond CCGs

Even though the first SENSEI's proof-of-concept was focused on CR, it could be easily extended to other CCGs, such as HS. Moreover, we used SENSEI to

[9] `https://activeplayer.io/clash-royale/`

[10] `https://docs.royaleapi.com/json/player_2JGYG2YY_battles.json`

[11] `https://activeplayer.io/hearthstone/`

[12] `https://kafka.apache.org/`

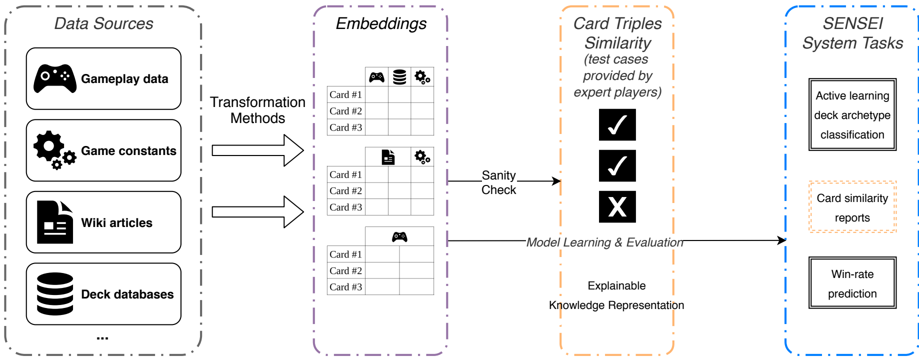

Figure 1: Visual representation of embeddings in the SENSEI framework. First, embeddings are created from data sources available for a particular game. Each embedding type is obtained by the transformation of one or multiple available data sources. The data sources usually define which approaches, therefore embeddings can be used for a particular task. The embeddings try to capture semantic differences between the objects in an ML-friendly representation. In SENSEI, as reported further in Section 5, they are first validated using card triples similarity task. This task is used as a sanity check if embeddings properly capture the domain knowledge in an explainable manner. This preliminary test can be also used to choose the best embedding size for an approach and domain if we generated embeddings with multiple sizes. At last, the chosen embeddings are used to train ML models solving the SENSEI's tasks.

create an advisory portal for a different video game, called *Tactical Troops: Anthracite Shift* (TT)[13]. TT belongs to a different type of video game genre than CCGs – it is a turn-based tactical combat simulation game. Nevertheless, almost all of the concepts discussed in this section can be translated into TT. This fact illustrates the universality of the SENSEI framework.

Instead of deck building, in TT there is a preparation of a so-called loadout for a game. It consists in choices made before a match regarding the starting resources, which is an analogy to a deck that a player brings to a match. In TT, this refers to unit types for the squad, as well as weapons assigned to them from the available pool. SENSEI can assist in the process of loadout creation exactly like it handles the deck creation, i.e., by suggesting items of equipment and units that will lead to the strongest combination due to the estimated win-rate criteria. TT's version of SENSEI predicts also the preferences of players and suggests them loadouts that will likely fit well to their playing style.

SENSEI also delivers in-depth statistics and aggregations for collected data. For TT, it additionally provides a special replay tool that allows players to watch previous games of their own or other players (e.g., the most skillful ones). It facilitates advanced annotation and displays statistics for each map. The statistics are shown in form of heatmaps displayed on top of the gameplay. The heatmaps inform, e.g., where troops usually die, where explosions usually occur (shown in Figure 2), or where grenades are thrown. Thanks to such information,

[13]https://sensei.tacticaltroops.net/

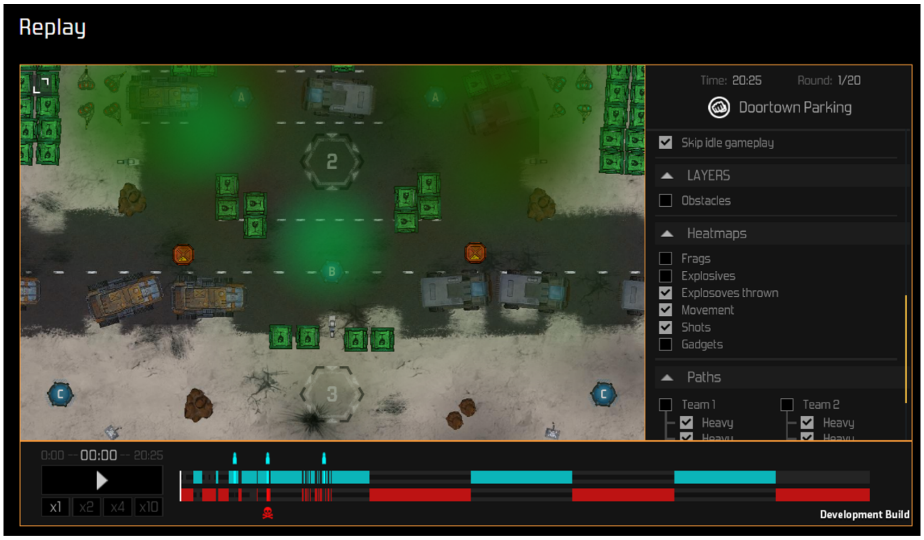

Figure 2: Replay tool in the SENSEI framework developed for Tactical Troops: Anthracite Shift. A particular heatmap is toggled on which shows the magnitude and range of explosions.

players become more aware of places that are particularly dangerous, learn where traps usually occur and where they should not leave their units.

Going further, it is often useful to measure (dis)similarity of in-game concepts and visually represent the results of measurements in form of clusters. In CCGs, we analyze the similarities of cards and decks and cluster them into archetypes, whereas for the TT purposes, we can cluster players into similar profiles, e.g., aggressive players. This is yet another example of a universal method of representation albeit applied to different game notions.

In summary, although in this paper we refer to CCGs as the main case study to examine the performance of our game concepts representation methods, the analogous methods of learning, combining and utilizing game entity representations can be used for other games. Moreover, we believe that the formulations of analytic tasks specific to the CCGs often have a universal meaning, and the proposed solutions can be transferred to other non-game domains.

4. Learning Card Embeddings from Multi-source Data

To learn representations that facilitate the analytics of CCGs, we need to cover a broad range of aspects related to considered games. We need to model complex dependencies between game rules and cards and capture many possible ways they can interact with each other. For this purpose, we need to take advantage of all available data sources. We start with the task of representing the building blocks of such games – the cards and card decks. Since the relationship between cards and decks bears some resemblance to the relationship of words and sentences, we explore the versatility of the NLP techniques to this end.

Herein, we aim to encode each card as a vector from multidimensional space, applicable in various prediction and visualization tasks. In further subsections, we will proceed with more advanced representations and finally, we will discuss how to combine them with each other to obtain efficient ML models.

More formally, let us consider a set of game entities U. We model the problem of constructing embeddings of instances from U, based on an available data source DS_U, by means of specifying the following function:

$$e_{DS_U} : U \longrightarrow R^m \tag{1}$$

such that the following implication holds:

$$(e_{DS_U}(u_1), e_{DS_U}(u_2)) \in \tau^{Sim}_{R^m \times R^m} \Longrightarrow (u_1, u_2) \in \tau^{Sim}_{U \times U} \tag{2}$$

where $\tau^{Sim}_{U \times U} \subseteq U \times U$ and $\tau^{Sim}_{R^m \times R^m} \subseteq R^m \times R^m$ are similarity relations defined over U and R^m, respectively. $\tau^{Sim}_{U \times U}$ can only be expressed indirectly by experienced players, e.g., by indicating pairs of cards or decks that they consider similar in the context of their function in a game. To approximate $\tau^{Sim}_{U \times U}$, we explore multiple data sources and embedding construction methods such as expert knowledge embeddings, text mining of community-made resources, learning embeddings from decks created by players, and analyzing in-game battles.

4.1. Expert knowledge-based embeddings

The most straightforward way of constructing card embeddings relies on human expert knowledge. Experts – typically experienced and skillful players – are often able to indicate the most important characteristics of considered entities. In case of CCGs where the number of cards is typically large (for HS, there are 1 596 collectible cards and, e.g., more than 13 600 cards in *Magic: the Gathering*), apart from defining relevant features of game cards, the experts could also provide a method for extracting their values from available game resources. This aspect is particularly important for games that evolve in time, e.g., by adding new cards with different mechanics. In such a situation, the experts need to be continually involved to monitor and update the resulting set of features. Fortunately, databases with card descriptions are publicly available and well-maintained for the most of popular eSport games. In particular, for the games considered in our study, there are specialized APIs[14][15] that allow us to obtain the most fundamental information about cards in a convenient way.

For the purpose of our study, we asked a group of experienced CR and HS players to define features for cards, based on information provided by the available APIs. As a result, we obtained card characteristics composed of a mixture of numeric (e.g., mana cost) and symbolic features (e.g., type of a card). In total, the experts defined 32 and 39 features for CR and HS, respectively.

[14]https://developer.clashroyale.com/
[15]https://hearthstonejson.com/

4.2. Text-mining embeddings

Another source of information about video games are fan-based web portals. Players often create Wikipedia-like sites for their favourite games, with separate articles devoted to individual game entities. For SENSEI, sites like that are of a great value. Portals related to CCGs have special sections about all cards from the game. Such articles often provide detailed information about card's basic characteristics, as well as natural language descriptions of their advanced properties and related in-game usage strategies. This data can be used to create card embeddings by employing common text-mining techniques.

In our experiments, we consider three different representations that make use of natural language descriptions of cards from Wiki pages. The first one is based on a standard *tf-idf* representation. The Wiki pages corresponding to particular cards are regarded as regular textual documents. After a typical text cleaning step (i.e., lowercasing, removing punctuation, and stemming), we compute the frequency of each term in the document and multiply it by the logarithm of inverted frequency of its occurrence in all card descriptions. As a result, for CR we obtained vectors of size 1 683, and for HS is was 5 636. The big difference in the vector size can be explained by the fact that the total number of cards available for HS is a few times higher than for CR. Indeed, in CR each card level has the same textual description. Thus, at the time of our experiments, the number of cards in HS was 1 595 vs. only 96 in CR.

The second representation extends the first one. We transformed the obtained card-term matrix using the Latent Semantic Analysis (LSA) technique [43] into a vector space of hidden concepts. We considered several sizes of the concept space (the number of concepts used in the embedding) and finally, after the analysis of eigenvalues, we decided to rely on vectors of size 50.

The last representation created from textual data was based on *word2vec* embeddings of words from the articles [36]. We used the card descriptions to train word embeddings using the skip-gram method. Since the vocabulary size of considered texts was relatively small, we decided to use much smaller word embedding size than in typical text mining applications. We experimentally checked several sizes between 10 and 50. We noticed that adding more dimensions to represent the terms brings negligible improvements in values of the loss function. To create the final card embeddings, we aggregated embeddings of terms from the textual descriptions by simply taking their mean.

4.3. Context-based embeddings

The third data source considered in SENSEI are specialized databases storing meta-data extracted from on-line games. For CCGs, such repositories contain information about game results along with descriptions of match-ups, e.g., players' IDs, their rankings, and decks that they used. Game developers collect this data to monitor in-game trends and fine-tune game balance. In many cases, they provide specialized APIs allowing players and content creators to freely access some of their resources. For instance, the API for CR allows to obtain data about 35 most recent games of each player. Similar repositories are created by the community. For instance, for HS there is the card and deck database maintained by the HearthPWN portal[16]. Nevertheless, data-driven SENSEI-like frameworks are easier to maintain when the game and its advisory portal are developed by the same company (like for TT in our case).

Data about deck compositions in actual games can be used to learn representations of cards and decks. One way of doing that is inspired by a word embedding learning method, i.e., continuous bag of words (CBOW) [35]. Each deck composed by a player can be considered as a sentence with words corresponding to cards. The ordering of cards can be imposed, e.g., by the order in which they were used during a game or in other arbitrary way. In our case, the representation is learned by training a simple neural network to recognize the n-th card from an n-card deck, based on information about a subset of the remaining $n - 1$ cards (the context) [18]. In this way, the resulting embedding of each card is such that its cosine similarity to the average of context embeddings is high. Noise contrastive estimation combined with negative sampling is a common choice for the model training [33]. This approach can be extended to incorporate additional information about decks, e.g., deck archetypes.

For the purpose of our study, we collected a vast amount of data about decks from CR and HS. For CR, we obtained meta-data from about 300 000 000 games played in the ranked mode from October 2018 – March 2019[17]. We extracted from this data information about decks constructed by players and we used this set to learn the CBOW-based card embeddings. For HS, we downloaded over 335 000 decks composed by players from the aforementioned *HearthPWN* portal. We considered only decks created for regular ranked games. As with CR, we used the CBOW model to learn the embeddings. In both cases, we experimented with various embedding sizes but eventually – just like in the case of embeddings in Subsection 4.2 – we decided to represent the cards by vectors of size 50.

4.4. End2end embeddings learned from gameplay data

The availability of gameplay data gives us an opportunity to consider the card and deck embeddings from a perspective of their influence on the game outcome. For instance, we gathered a data set of 160 000 000 1v1 CR rank games using SENSEI's data acquisition module. Such an abundance of data allows us to employ more sophisticated ML approaches, whereby card embeddings can be trained along with a prediction model and they can be re-used later in other analytical tasks. In this subsection, we describe in detail one of such ML mechanisms which is actually one of significant novelties of our research. On the other hand, let us note that in this case we are interested not only in the efficiency of the resulting ML model itself, but also in the embedding representation that

[16] `https://www.hearthpwn.com/`

[17] This data set was used in a data science competition organized during the preliminary phase of our research [17]. It is still publicly available at the *KnowledgePit* platform `https://knowledgepit.ml/` providing the means for validating and extending reported experiments.

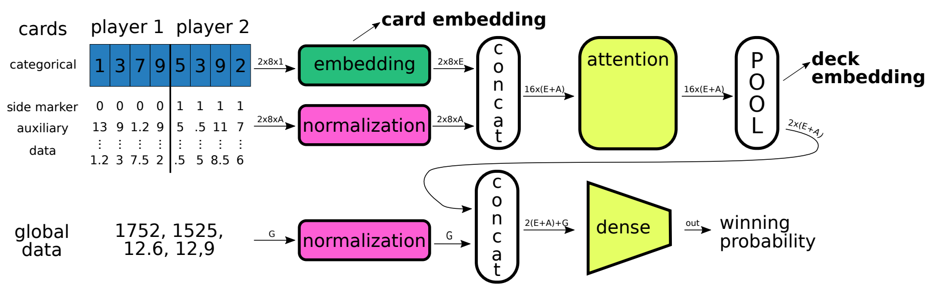

Figure 3: DNN architecture of our *end2end* model for CR (8 cards in deck). E is the size of embedding, A is the number of auxiliary information per card (e.g., level in CR), G is the number of global information (e.g., trophies of players in CR). The *attention* layer may be any of the described attention versions. It may focus only on a current player thanks to the side marker in the input. The pool layer makes the model order-independent by averaging the attention layer output over cards in a given deck. This vector may be interpreted as *deck embedding*, if the attention layer can only reference cards from the same deck.

it produces as "a side effect". This representation will be used, together with those discussed in Subsections 4.1-4.3, to compare the investigated ensemble construction strategies, as described in Subsection 4.5.

The task for the considered ML mechanism was to train a model that foresees which player is going to win the game based on available meta-data, such as the composition of players' decks and their current rankings. To this end, we decided to design and use a brand new neural network model with the attention mechanism inspired by [44]. Its architecture is depicted in Figure 3. First, categorical variables representing card IDs are projected to an E-dimensional real space by the embedding layer. Then, an initial internal representation of cards is built by concatenating the embeddings from the previous layer with the normalized auxiliary card data extracted from a battle (A features) (e.g., card levels for CR). The additional features have a positive indirect effect on the attention values and trained card embeddings as they provide a meaningful context (e.g., in games such as CR, some synergies between cards are visible only on certain card levels). The resulting representation is flattened to a $2n \times (E + A)$ matrix of the battle, where n in the number of cards in a deck. This matrix is further processed by an attention block described below. It computes altered representations of the cards in the context of the current game, while preserving the matrix shape. Since the order of cards is not relevant (decks are shuffled in the beginning of a battle), a pooling layer reduces the matrix shape to $2 \times (E + A)$, which could be interpreted as a representation of the decks in the battle. Finally, the features extracted from global game data (e.g., players' ranking) are appended and dense layers are used to predict the winner.

The purpose of the attention block is to infer interactions between cards. We applied this mechanism at the following four complexity levels: a single Luong-style attention [32] layer (*Pure attention*), an attention layer with four heads limiting the scope of attention to support and adversary pairs (*Supp-adv attention*), a single multihead attention layer (*Multihead attention*), and the full three layers of the *Transformer* encoder, as defined as follows:

$$Attention(Q, K, V) = softmax\left(QK^T/\sqrt{E+A}\right)V \quad (3)$$

$$Pure\_attention(X) = Attention(W^Q X, W^K X, W^V X) \quad (4)$$

$$Multihead\_attention(X, h) = \begin{bmatrix} Attention(W_1^Q X, W_1^K X, W_1^V X) \\ \dots \\ Attention(W_h^Q X, W_h^K X, W_h^V X) \end{bmatrix} W_0 \quad (5)$$

$$Supp\_adv\_attention(X, Y) = \begin{bmatrix} Attention(W_s^Q X, W_s^K X, W_s^V X) \\ Attention(W_a^Q X, W_a^K Y, W_a^V Y) \\ Attention(W_a^Q Y, W_a^K X, W_a^V X) \\ Attention(W_s^Q Y, W_s^K Y, W_s^V Y) \end{bmatrix} \quad (6)$$

whereby all W are learnable parameters. X in (4) and (5) is the battle-representing $2n \times (E + A)$ matrix, which is split in (6) into X, Y per-player $n \times (E+A)$ matrices. Additionally, in (5) there are h parts – so-called attention heads – which are finally put together using a learnable matrix W_0.

In our initial experiments on CR data, the first *Pure attention* model was able to learn counter-plays, i.e., cards attended (i.e., the respective cells of QK^T in (3) dominated a row, which correspond to cells of $W^Q X(W^K X)^T$ in *Pure attention* architecture) mostly to their counters, with the attention layer properly focusing on the enemy's deck (Figure 4). Further tests showed, however, that this behaviour was not stable, as it sometimes learned card synergies, i.e., two cards often played together. This issue was related to having only one attention head. The *Supp-adv* attention model learns both counter-plays and synergies in the respective slices. The *Multihead* and *Transformer* models had even better accuracy metrics, but the produced attention matrices were not as easily interpretable as the former. The best model, trained for the aforementioned task of predicting game-winners, achieved the accuracy of 64.4% for CR when provided with additional information modifying the card's behavior (i.e., card level).

Apart from individual card embeddings, the *end2end* approach allowed us to learn the embeddings of whole decks. We achieved that by employing the aforementioned attention architecture: a shared *Multihead* attention layer limited to attend only to the same deck. After aggregation between cards, we extract weights from this layer as deck embeddings (the pooling layer in Figure 3).

4.5. Combining multiple data representations

The presence of many different data sources and methods for constructing card and deck embeddings allows for modeling various aspects of CCGs. Then, when solving a specific analytical problem in SENSEI, one may want to focus on a particular aspect and the corresponding representation of data. However, it is important to choose the right representation, which in practice is not an easy task. Depending on a problem, some embedding types may prove more useful than others – to choose the best one, it is necessary to prepare an appropriate (labeled) data set and perform a quantitative evaluation. Since it is difficult

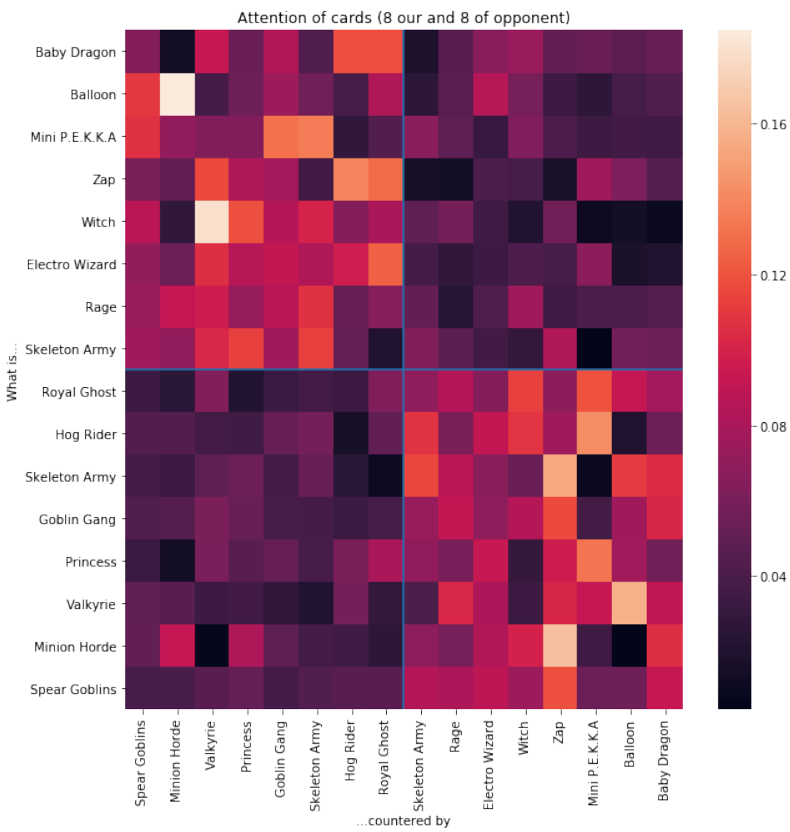

Figure 4: An attention map between cards in a CR battle as seen by the pure attention model. Attention between player 1 to player 2 cards is seen in the top-left quadrants, and the opposite in bottom-right. Many highlighted cells correspond to a counter-play – for instance Balloon is neutralized by Minion Horde. Not all most-attended cards are simple counters though – opponent Valkyrie is attending to Balloon, and those cards have no interaction, but the latter may be seen as a punishment-play for mana investment in the former.

to directly assess the quality of embeddings, a common approach is to evaluate their usefulness by measuring the performance of ML models that use them.

A viable alternative to selecting the best data embedding for a given task can be constructing a representation that combines various aspects of the data and allows the ML model to select those which are useful. During our initial study, we noticed that concatenating multiple embeddings of HS cards achieved higher accuracy than the individual representations for the problem of win-rate prediction [20]. In this paper, we further explore this idea by investigating two basic strategies of using an ensemble of data representations, i.e., concatenating embedding (called *concat* later on), and voting between models trained on data represented by the individual embedding types (called *voting*).

The above two strategies can be combined in various ways. For instance, the model can be independently trained on each representation, including the concatenation of the individual embedding types, and the final decision can be made using majority voting (for classification task) or averaging (for regression or similarity comparison task). We will refer to this idea as to *concat+voting*. As another example, various embeddings can be used to train simple models in a model ensemble, e.g., individual trees in a random forest or XGBoost model can be trained on data represented by a single randomly selected embedding type (but different trees must use different embeddings). In the description of our experiments, we will refer to this approach as a *constrained* ensemble.

The above strategies of utilizing multimodal data in the process of ensemble learning reflect different approaches to the construction of robust ML models. It is often emphasized in the literature that the diversity of features utilized by particular models in the ensemble may be important for its accuracy, stability, and applicability in case of temporary data incompleteness [38]. However, not so many solutions pay additional attention to the origin of those features and shift the study of their diversity toward the level of the underlying multimodal data sources [45]. Experiments in Section 5 show that the diversification of ML models with respect to embeddings that constitute their input spaces can pay off. In particular, we will see that a combination of the *concat* and *voting* strategies often leads to the best accuracy of ML ensembles.

5. Experimental Evaluation of Embedding Learning Methods

In this section, we describe the results of experiments that we designed to evaluate the usefulness and soundness of the data representations discussed in Section 4. Our focus on the attention-based representations introduced in Subsection 4.4 and the discussed strategies of operating with multimodal embeddings while training ML ensembles. We check the performance of ML methods in solving various prediction tasks considered in SENSEI. In particular, we consider the already-discussed CR and HS as two popular competitive games for which rich public data repositories and information resources are available. However, we believe that the considered representation methods can be applied to a much broader spectrum of problems. The data for constructing embeddings of in-game entities in CR and HS come from various sources. Each source focuses on a different aspect of the game, and thus may contribute to various analytical tasks. Table 1 outlines the data used in experiments. To evaluate the usefulness of embeddings and their combination methods, we employed these data sources to define three evaluation tasks. We denote them as Task I: card similarity test; Task II: win-rate prediction; Task III: deck archetype classification. Table 2 summarizes them briefly.

5.1. Task I: Card similarity test

Herein, we check if the considered card embeddings can capture non-trivial interactions and similarities between cards, which typically requires some deeper insights about the game. As visible in Figure 1, this task can be treated as a

Table 1: Summary of all data types related to CR and HS available for the experiments. Each of the considered data sources can be used to create embeddings of cards and decks.

Data type	**CR**	**HS**
Card-related data	Text descriptions and meta-data for 96 cards available through official API	Text descriptions and meta-data for 1 595 cards available through community portals
Gameplay meta-data and logs	Meta-data from 160 000 000 ranked 1v1 games, game logs unavailable	Logs and meta-data from 485 000 emulated bot games (on 340 most popular cards)
Player-made deck collections	Over 130 000 decks extracted from gameplay data, their win-rates estimated based on real game outcomes	Over 200 000 decks from community databases, win-rates estimated for 600 decks using games between bots

Table 2: Summary of evaluation tasks and sizes of corresponding data sets.

Task id	**Task name**	**Overall data set size (CR | HS)**	**Task type**
Task I	Card similarity test	100 | 100	binary
Task II	Win-rate prediction	130 223 | 2 400	regression
Task III	Deck archetype classification	1 000 | 1 000	multi-class

kind sanity check prior to considering more advanced analytical goals. The examined collections of card triples are acquired from expert players. A part of this data has been already used in our previous research [18]. However, for this study, we collected a number additional card triples from CR and we exchanged some of the HS cards to reflect recent updates to the game. The used card triples and their labels are included in Supplementary Materials.

Formally, we consider Task I as a verification of whether the similarity between cards in the embedding space (the aforementioned $\tau^{Sim}_{R^m \times R^m}$) is consistent with semantic similarity perceived by expert players ($\tau^{Sim}_{U \times U}$). Given a set of instances U, a data source DS_U, and a similarity measure Sim, we wish to construct an embedding function e_{DS_U} that maps each object $u \in U$ into an embedding space R^m in a way such that equation (2) is valid. For evaluation, we use the card triples obtained from experts as a set of test questions C. Each question $c \in C$ consists of three instances $c_1, c_2, c_3 \in U$ and a binary answer $I \in \{TRUE, FALSE\}$ indicating whether $Sim(e_{DS_U}(c_1), e_{DS_U}(c_2))$ should be greater than $Sim(e_{DS_U}(c_1), e_{DS_U}(c_3))$. The quality of function e_{DS_U} is expressed as the accuracy of answers to questions from C, i.e., $Acc(e_{DS_U}) =$

$$= \frac{\sum_{(c_1,c_2,c_3;I)\in C} (Sim(e_{DS_U}(c_1), e_{DS_U}(c_2)) > Sim(e_{DS_U}(c_1), e_{DS_U}(c_3)) == I)}{|C|} \tag{7}$$

We use the cosine similarity metric in the experiment since it is a common choice when working with high-dimensional data, however, other functions could also be used. Following the card similarity task specification, we evaluate the previously created card embeddings by computing the consistency of similarity values in the embedding space with the labels indicated by experts. Tables 3 and 4 outline the evaluation results. For each embedding described in Section 4, we report the value of $Acc(e_{DS_U})$ defined by equation (7), as well as the size of e_{DS_U}, i.e., the number of its dimensions. Apart from embeddings corresponding to individual data sources, we include results for the ensemble strategies discussed in Subsection 4.5. *Concat* represents the concatenation of all embeddings excluding the bag-of-words (it was omitted due to overly high dimensionality), whereas *voting* corresponds to the majority voting between similarity indications obtained from the individual embeddings. We also report results for the *concat+voting* algorithm, i.e., the majority voting with the *concat* representation added as one of the voters. Since in this case, the number of voters was even, draws were resolved in favor of the pair for which the sum of cosine similarities computed using the voting embeddings was greater.

5.2. Task II: Prediction of deck win-rates

In this task, we checked if the previously created embeddings of cards are useful for representing whole decks and whether they can facilitate the construction of models for predicting deck win-rates. This task is particularly important for the SENSEI framework – we would like to estimate the quality of decks to advise the most effective compositions to our users. The win-rate of a deck is a percentage of games won by players with this deck. It is the main indicator whether a particular card selection is advantageous. Usually, in a well-balanced game, the best deck win-rates oscillate around 50%, while some poorly selected decks may have a win-rate as low as 0%.

Formally, Task II can be defined as a regression problem. Given a set of instances U and a data source DS_U, we divide it into three parts, i.e., training U_{tr}, validation U_{val} (for hyper-parameter tuning), and test part U_{te} (for the evaluation). We aim to create a prediction model M, trained on various data representations $e_{DS_U}(U_{tr})$, with the highest generalization quality measured on $e_{DS_U}(U_{te})$. We quantify the generalization quality using a regression quality metric. In our experiments, we use the determination coefficient R^2:

$$R^2 = 1 - \frac{\sum_{u\in U_{te}} (M(e_{DS_U}(u)) - \bar{y})^2}{\sum_{u\in U_{te}} (y_u - \bar{y})^2} \tag{8}$$

where y_u is the actual win-rate of $u \in U_{te}$, $M(e_{DS_U}(u))$ is a prediction of M for this instance, and $\bar{y}$ is the mean y_u over all instances from U_{te}.

Table 3: Summary of the embedding similarity evaluation results for CR.

Embedding	**Summary**	**size**	***Acc***
concat+voting	Ensemble of concat, CBOW, end2end, LSA, word2vec, expert (Subsection 4.5)	192	0.87
voting	Ensemble of embeddings CBOW, end2end, LSA, word2vec, expert (Subsection 4.5)	192	0.86
concat	Concatenation of CBOW, end2end, LSA, word2vec, expert (Subsection 4.5)	192	0.80
end2end	Embeddings trained using the *end2end* approach (Subsection 4.4)	50	0.76
CBOW	The *CBOW* model trained on a deck collection (Subsection 4.3)	50	0.76
expert	Embeddings proposed by a human expert (Subsection 4.1)	32	0.72
LSA	*LSA* embeddings build on textual data (Subsection 4.2)	50	0.71
word2vec	The *skip-gram* model trained on card descriptions (Subsection 4.2)	10	0.70
bag-of-words	The *bag-of-words* from textual descriptions (Subsection 4.2)	1 683	0.67

The data is characterized in Table 1. For CR, we used a data set from the aforementioned open ML competition [17]. The deck win-rates were estimated based on data from 160 000 000 PvP ranked games played between October 2018 and March 2019. The training set was composed of the most popular 100 000 decks in the first three game seasons, and the test set contained another 24 223 decks used in the following seasons. There was also available a validation set of 6 000 decks from the same period as the test data. For HS, we used the data from the ML competition [20]. In this case, however, the data was limited to 600 popular decks, whose win-rates were estimated by simulating 485 000 games between four different types of AI bots. The bots were based on the MCTS algorithm and were described in more detail in [20]. Using the simulations was necessary due to a lack of possibility to acquire a sufficiently large number of HS game results. For both games, the division of data and used evaluation metrics were exactly the same as in the corresponding competitions.

Our goal was to verify the influence of input representations on the learning capability of various ML models. We created deck embeddings by averaging the embeddings of the corresponding cards and concatenating the resulting vectors with the bag-of-cards representations. Additionally, we directly learned deck representations using the *end2end* approach (Subsection 4.4). We did it by taking embeddings learned by the corresponding attention-based model which was trained to predict winners in games, based only on the compositions of the involved decks. For each of embeddings, we trained three commonly used pre-

Table 4: Summary of the embedding similarity evaluation results for HS.

Embedding	Summary	size	*Acc*
concat+voting	Ensemble of concat, CBOW, end2end, LSA, word2vec, expert (Subsection 4.5)	239	0.83
voting	Ensemble of CBOW, end2end, LSA, word2vec, expert (Subsection 4.5)	239	0.78
concat	Concatenation of CBOW, end2end, LSA, word2vec, expert (Subsection 4.5)	239	0.75
expert	Embeddings proposed a human expert (Subsection 4.1)	39	0.72
word2vec	The *skip-gram* model trained on card descriptions (Subsection 4.2)	50	0.72
CBOW	The *CBOW* model trained on a deck collection (Subsection 4.3)	50	0.69
bag-of-words	The *bag-of-words* from textual descriptions (Subsection 4.2)	5 636	0.66
LSA	*LSA* embeddings build on textual data (Subsection 4.2)	50	0.65
end2end	Embeddings trained using the *end2end* approach (Subsection 4.4)	50	0.59

diction models: generalized linear models with the ridge regularization (*GLM*), k-Nearest Neighbors (*kNN*), and XGBoost [7] (*XGB*). We expect this selection to show the generality of our multimodal representations. We did not focus heavily on the hyperparameter fine tuning. For GLM, we used the implementation from the *glmnet* library, with *alpha* set to 0.0 and *nlambdas* set to 200. For XGB, the learning rate *eta* was set to 0.01, the number of trained trees was 1 000, and the maximal tree depth was 10. Additionally, heavy L_1 and L_2 regularization was used. For kNN, the k parameter was set to 53 and the *Gaussian* kernel was used for neighbours weighting.

5.3. Task III: Classification of deck archetypes using active learning

Deck archetypes can be seen as a categorization of decks by the collective community of players. The prediction of deck archetypes is therefore a classification problem. An archetype often defines the gameplay strategy of a deck. Players usually have a preferred style of gameplay, thus recommendations of card substitutions in a deck should preserve its archetype. This feature is especially valuable for newbies who are accustomed to only a few decks but want to expand their horizons. SENSEI, thanks to deck classification capabilities, can suggest similar decks with new cards that still fit to the same archetype. Thus, the players can improve their skills in using new cards and at the same time can use the same strategies that they are familiar with. Unfortunately, information about deck archetypes is often not available in the community-managed deck repositories, or its quality is questionable, e.g., there is no fixed taxonomy allowing to label archetypes in a consistent way. For this reason, we consider the problem of learning archetypes through the interaction with players, i.e., we take the active learning approach [40] which allows us to construct efficient archetype classification models using relatively small subsets of labeled decks.

Formally, we define Task III as a limited training data pool-based active learning problem. Given an instance pool U_{tr}, data source DS_U, the corresponding training data representations consisting of records $(e_{DS_U}(u), y_u)_{u \in U_{tr}}$, and a prediction model M, we search for a training subset U_{opt} of a fixed size N, such that the model trained on this subset achieves the highest generalization quality measured on hold-out sets $(e_{DS_U}(u), y_u)_{u \in U_{te}}$, i.e.:

$$U_{opt} = \arg\max_{U^* \subseteq U_{tr}: |U^*| = N} \mathbb{E}_{possible\, sets\, U_{te}} q\left((M^{U^*}(e_{DS_U}(u)), y_u)_{u \in U_{te}} \right) \quad (9)$$

where M^{U^*} is the mapping induced by the model M trained on the subset U^*, $\mathbb{E}$ is the expected value over a space of possible hold-outs, and q is a quality metric defined over sets of hold-out tuples $(M^{U^*}(e_{DS_U}(u)), y_u)_{u \in U_{te}}$. In our experiments, q is the balanced accuracy measure (bac).

In our approach, we begin by choosing a smaller initial training subset at random. Then, we utilize the algorithm based on density weighted batch uncertainty sampling [40]. Specifically, at each step, we select an instance that maximizes the importance:

$$u_{opt} = \arg\max_{u \in U_{tr}} \left[\phi(u)^{\alpha} \cdot Sim(u)^{\beta} \cdot Dis(u)^{\gamma} \right] \quad (10)$$

where ϕ is a measure of the informativeness of instances,

$$Sim(u) = \frac{1}{|U_{tr}|} \sum_{u' \in U_{tr}} sim(u, u') \quad (11)$$

measures their similarity-based representativeness, and

$$Dis(u) = \frac{1}{|current\, batch|} \sum_{u'' \in\, current\, batch} dis(u, u'') \quad (12)$$

measures the dissimilarity in the current batch, assuming that we have already chosen some instances u''. Parameters α, β, γ control the relative importance of each factor. In our experiments, they were set to $\alpha = \beta = \gamma = 1$. As about ϕ, we employed the entropy of estimated class probabilities returned by the prediction model at a given stage of the process. Let us note that in our formulation of the batch selection problem, the notion of similarity between instances is of paramount importance. In particular, we set $dis(u_1, u_2) = 1 - sim(u_1, u_2)$, so the two last terms in equation (10) depend on similarity values.

To assess the importance of card embeddings for the batch selection process, we evaluate the overall performance of the archetype prediction model during the active learning cycle for the deck similarities induced by different sets of card embeddings. As in Task II, we compute embedding of a deck as a mean of its card embeddings and we use the cosine similarity measure. To implement the voting strategy for the representation ensembling, we rank similarities computed for each individual embedding, and we average the ranks. (We scale the similarity matrices to the 0-1 range by dividing by the maximum value.) We also checked an alternative voting strategy – called *representatives* – in which the instances for a new batch are ranked with regard to each individual embedding, and the batch is selected by sequentially taking top-ranked instances in each ranking (in random order) until the required batch size is obtained.

We compare the obtained active learning results with the results achieved using the bag-of-cards representation, and also to the baseline approach where (dis)similarity measures are not taken into account (i.e., $\beta = \gamma = 0$). In our experiments, we use a logistic Lasso model M, the data pool set size equal to 700 and the initial training subset size equal to 100. We finish the active learning procedure when the total number of instances selected for model training is 300. We train the model only on the bag-of-cards representation, as our main goal is to highlight the impact of embeddings on similarity-based instance selection. For each experiment, we evaluate the active learning procedure for 10 random pool-test splits and for each split, we perform 25 independent runs for randomly sampled initial training subsets. Thus, for each embedding type, we perform 250 runs in order to provide reliable quality evaluation estimates.

6. Analysis of the Experimental Results

The results obtained for the first of considered tasks confirm our main research claims. For both games, the combinations of different types of representations led to considerably better models than any of the individual embeddings (see Tables 3 and 4). Moreover, in both cases, the voting strategy brought a greater improvement than the concatenation of embeddings, and *concat+voting* strategy obtained the best outcomes. At the same time, there seems to be no pattern regarding the quality of individual representations. It shows that in practical applications it is difficult to select in advance a single data source and a representation that can capture the similarity between game entities.

Another observation refers to the proposed *end2end* approach. For CR, it turned out to be the best individual embedding to measure the similarity between cards, while for HS it was the worst one. This is most likely related to the data availability issue. In case of CR, the neural network was trained on a large collection of real game outcomes, with additional information about the context of each game (i.e., player rankings and card levels). For HS, the model was trained only on games between bots, and the number of available games was significantly lower (see more details in Table 1). It is aligned with common experience that approaches based on neural networks do require significant volumes of training data and, if such volumes are not in place, it is better to rely on simpler methods. On the other hand, to demonstrate their usefulness for both games, we measured the impact of removing *end2end* embeddings from the best-performing ensemble method. For both games, we noticed a drop in the accuracy value, i.e., for CR it dropped from 0.87 to 0.81 (0.06 change), while

Table 5: Results (values of the R^2 metric) obtained for the deck win-rate prediction tasks for CR and HS. Data representations corresponding to each embedding type were used to train three models: GLM, XGBoost, kNN. The *concat* column shows results obtained by using concatenation of all embeddings; *voting* corresponds to results of an ensemble of models trained on individual embeddings; *concat+voting* shows results of a combination of the *voting* and *concat* representation ensembling strategies, i.e., an ensemble of models trained on the *concat* and the individual embeddings (see Subsection 4.5). The baseline results correspond to plain bag-of-cards representations of decks. e2e stands for *end2end* (Subsection 4.4).

For CR	**base-line**	**expert**	**CBOW**	**card e2e**	**word-2vec**	**LSA**	**deck e2e**
GLM	0.1300	0.1304	0.1306	0.1305	0.1305	0.1313	0.1712
XGB	0.2565	0.2599	0.2549	0.2607	0.2614	0.2616	0.2601
kNN	0.2207	0.2264	0.2222	0.2226	0.2280	0.2163	0.2240

For HS	**base-line**	**expert**	**CBOW**	**card e2e**	**word-2vec**	**LSA**	**deck e2e**
GLM	0.6877	0.6983	0.6939	0.6655	0.6896	0.6989	0.6790
XGB	0.5962	0.6285	0.6092	0.6355	0.6264	0.6281	0.6605
kNN	0.3767	0.3755	0.3762	0.3741	0.3775	0.3763	0.4180

For CR	**concat**	**voting**	**concat voting**	**For HS**	**concat**	**voting**	**concat voting**
GLM	**0.1731**	0.1394	0.1455	GLM	0.6865	0.6985	**0.7001**
XGB	0.2663	0.2691	**0.2701**	XGB	0.6441	0.6944	**0.6997**
kNN	0.2247	0.2304	**0.2307**	kNN	**0.4187**	0.3836	0.3890

for HS it dropped from 0.83 to 0.82 (0.01 change). It shows how beneficial it is to combine representations obtained from different data modalities.

The outcomes of the tests conducted for Task II lead to similar conclusions. The results clearly show the advantage of the multimodal representation ensembling strategies over the usage of individual embeddings. Both *concat* and *voting* approaches allowed us to achieve better predictions in terms of the *determination coefficient* (the R^2 metric) than the baseline bag-of-cards representation, as well as the most of embeddings based on a single data source. When it comes to our *deck-end2end* representation, it was better than the voting ensembles for the GLM model on CR data. It also achieved a better result than *concat* for the XGB model for HS. Still, even though we expected that approach to yield good results, it proved to be less efficient than using the ensemble of models trained on each of the considered representations. In some cases, the evaluation score of models trained on *end2end* embeddings was slightly lower than scores obtained for textual descriptions of cards (e.g., for XGB trained on the CR data). Moreover, for both games, the aggregations of card-level *end2end* embeddings were worse than the *end2end* embeddings obtained for whole decks.

Similarly to Task I, we checked the influence of removing the *end2end* embeddings from the best-performing ensembles. Again, we noted a considerable drop in predictions quality. In particular, for CR the quality dropped from 0.1731 to 0.1367 for GLM, from 0.2701 to 0.2594 for XGB, and from 0.2307 to

Table 6: Results for different embedding-based representativeness and dissimilarity measures in active learning of deck archetype classification for CR. The quality of each embedding is measured using the mean *bac* over all pool-test splits, initial training random subsets, and training sizes. 5, 15, 25 are the batch sizes. A total of 250 runs of the whole experiment for each embedding type and batch size. Std indicates a standard deviation of the results.

Embeddings used in AL	**5 mean**	**5 std**	**15 mean**	**15 std**	**25 mean**	**25 std**
random: $\beta = \gamma = 0$	0.9108	0.0059	0.9099	0.0053	0.9100	0.0057
bag-of-cards	0.9187	0.0059	0.9185	0.0059	0.9182	0.0058
expert	0.9211	0.0049	0.9208	0.0046	0.9200	0.0045
CBOW	**0.9256**	0.0047	**0.9236**	0.0049	**0.9223**	0.0048
card-end2end	0.9251	0.0050	0.9231	0.0054	0.9218	0.0053
word2vec	0.9207	0.0054	0.9202	0.0050	0.9193	0.0051
LSA	0.9167	0.0066	0.9183	0.0063	0.9185	0.0060
deck-end2end	0.9162	0.0063	0.9179	0.0058	0.9175	0.0062
concat	0.9163	0.0072	0.9153	0.0075	0.9146	0.0073
voting	0.9203	0.0052	0.9209	0.0052	0.9204	0.0052
concat+voting	0.9184	0.0062	0.9185	0.0059	0.9184	0.0057
representatives	0.9221	0.0065	0.9207	0.0066	0.9196	0.0065

0.2218 for kNN. For HS, the R^2 metric values dropped from 0.7001 to 0.6917 for GLM, from 0.6997 to 0.6576 for XGB, and from 0.4187 to 0.3758 for kNN. Finally, a detailed comparison of the results for *concat* and *voting* methods shows that usually the latter one allows to obtain better results. Nevertheless, in a majority of cases, the combination of those two strategies (*concat*+*voting*) yields the best overall results, which suggests that this is the most robust approach for the win-rate prediction task. Table 5 shows the detailed results obtained for each embedding and each multimodal representation ensembling strategy.

For Task II, we also conducted an additional experiment to check if various embeddings can facilitate training base models in an ensemble learning algorithm such as the gradient boosting. For this purpose, we modified the code of the XGBoost library to change the way in which individual trees are trained, i.e., we limited the selection of features for the construction of each tree to a single randomly chosen embedding type, and we allowed that each tree used a different embedding. The probability of selecting any particular type of representation, including the concatenation of all base embeddings, was equal. This is actually an XGBoost-specific implementation of the *constrained* approach mentioned in Subsection 4.5, whereby *concat* is used as an extra embedding. We observed that for HS, this approach achieved a considerably better R^2 value than the XGB model trained on the concatenation of all embeddings ($0.6441 \longrightarrow 0.6725$) while for CR, the results were comparable ($0.2663 \longrightarrow 0.2649$). This shows that such strategy can also be feasible and should be investigated in future.

Tables 6 and 7 show the averaged results of experiments for Task III, obtained for CR and HS, respectively. The performance was assessed using the *bac* measure. All embedding methods outperformed the random selection baseline

Table 7: Results for different embedding-based representativeness and dissimilarity measures in active learning of deck archetype classification for HS. Quality measured like in Table 6.

Embeddings used in AL	**5 mean**	**5 std**	**15 mean**	**15 std**	**25 mean**	**25 std**
random: $\beta = \gamma = 0$	0.9806	0.0036	0.9789	0.0028	0.9764	0.0031
bag-of-cards	0.9881	0.0051	0.9850	0.0048	0.9821	0.0045
expert	0.9866	0.0070	0.9844	0.0052	0.9821	0.0043
CBOW	0.9873	0.0064	0.9839	0.0049	0.9806	0.0049
card-end2end	0.9839	0.0069	0.9806	0.0071	0.9781	0.0066
word2vec	0.9855	0.0062	0.9831	0.0053	0.9804	0.0047
LSA	0.9853	0.0063	0.9826	0.0058	0.9798	0.0057
deck-end2end	0.9868	0.0037	0.9834	0.0033	0.9802	0.0030
concat	**0.9885**	0.0036	**0.9858**	0.0033	**0.9826**	0.0029
voting	0.9870	0.0062	0.9845	0.0054	0.9817	0.0050
concat+voting	0.9872	0.0059	0.9845	0.0054	0.9814	0.0053
representatives	0.9876	0.0046	0.9841	0.0041	0.9809	0.0040

for both games. For CR, the best results were obtained by the *CBOW* embeddings. We checked if the differences between results for *CBOW* and *bag-of-cards* are statistically significant, and we were able to reject the null hypothesis about equality of the means using the paired t-test with *p-value* < 0.001 for all considered batch sizes. For HS, the best results were obtained by the representation ensembling methods (i.e., the *concat* embeddings). In this case, however, the second best results were for the standard *bag-of-cards* representation, and the differences between the best two embeddings were not statistically significant (*p-value* > 0.1). The reason for this may be the fact that the problem of learning deck archetypes for HS turned out to be quite simple – all considered embedding methods achieved high *bac* values. Figures 5 and 6 illustrate the averaged results for each embedding type in consecutive iterations of the active learning cycle when the batch size was set to 5. The corresponding plots for the batch size 15 and 25 are added to Supplementary Materials.

The representation ensembling approaches, especially those which are based on a voting strategy, proved to be reliable choices for both CR and HS. Although for CR the *CBOW* and *card-end2end* embeddings obtained slightly better results, *voting* and *representatives* (see Subsection 5.3) were not far behind. Their results could also be considered significantly better than for the baseline (i.e., $\beta = \gamma = 0$), with *p-value* < 0.05 for all considered batch sizes. Interestingly, the *concat* approach did not work well for CR. It shows that using such a high-dimensional representation for the computation of similarities can be risky.

7. Conclusions

Data representation is ever-present in AI, ML, and computer science in general. When developing intelligent systems, feature engineering is one of the

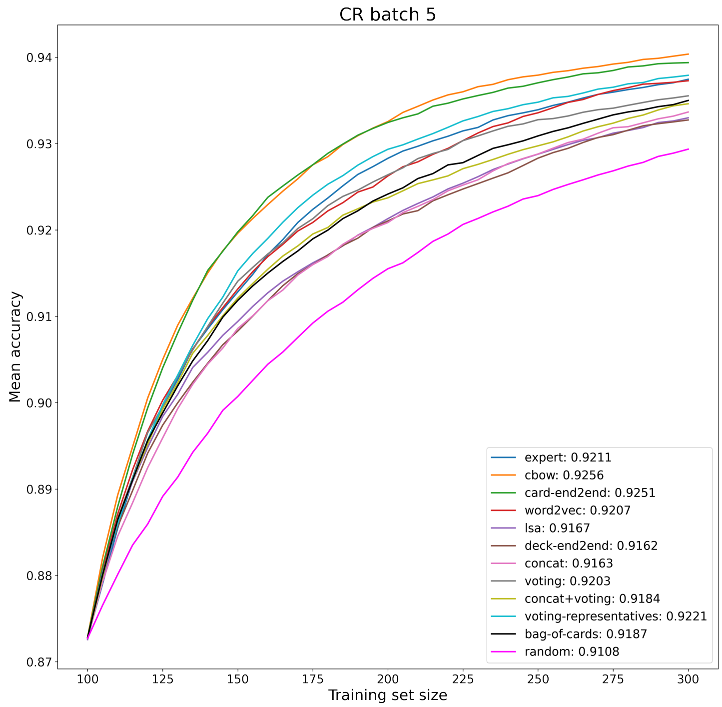

Figure 5: Average results obtained by various embedding types in consecutive iterations of the active learning of CR deck archetypes when the batch size 5 was used.

most pertinent, time-consuming and ambiguous tasks. Although it is typically carried out in the early stages of development, it has a huge impact on the final performance of a system. In this article, we focused on multimodal data often available in the video game domain. In our case study, we considered collectible card games (CCGs) – Clash Royale (CR) and Hearthstone: Heroes of Warcraft (HS), whereby data sources correspond to card descriptions, game logs, expert knowledge provided by players, web portals, forums, results of simulations, etc. However, we believe that our findings generalize to other video games, as well as many real-life applications which can go far beyond the video game industry.

We comprehensively studied various approaches to entity representation learning, which are based on both, standard techniques from the NLP domain and our own novel ideas. We investigated the possibility to represent game cards by means of their textual descriptions – as bags-of-words, using LSA, or by aggregated word2vec embeddings. We also used the CBOW model to train the embeddings based on large collections of player-made decks. On the other hand, we introduced a neural network architecture to learn embeddings based on the outcomes of real games. What is even more important, we verified the useful-

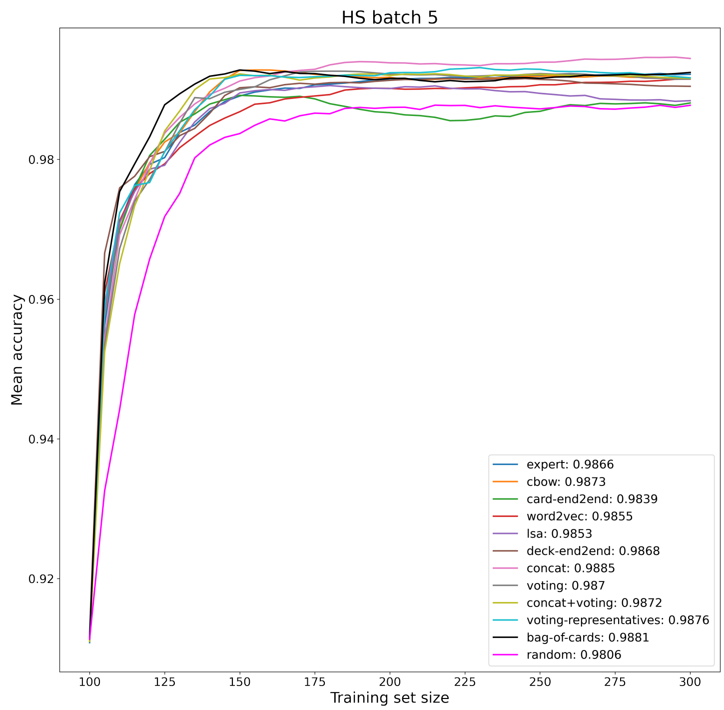

Figure 6: Average results obtained by various embedding types in consecutive iterations of the active learning of HS deck archetypes when the batch size 5 was used.

ness of embeddings ensembles, i.e., the ensembles of ML models trained using different configurations of data representations. We considered several strategies for combining different representations, whereby some of them were not thoroughly investigated in the literature so far. In particular, we demonstrated the usefulness of ensembles whereby some ML models rely on single embeddings while the others browse through multimodal embedding concatenations.

In experiments, we confirmed that all of the considered representations can be useful. We checked their performance in several practical problems of video game analytics, i.e., detecting similarities between entities (Task I), estimating win-rates of decks (Task II), and discovering deck archetypes using the active learning approach (Task III). We demonstrated that the ensembles in which independent models are trained on individual embeddings are typically the safest choice and usually obtain the best results. It is also beneficial to include a model trained on a concatenation of the embeddings corresponding to different data modalities as one of the voters in the final ensemble. This particular approach outperformed other methods for both CR and HS in two of the test problems (Tasks I and II). Moreover, in case of Task II, the performance increase was noticeable for two out of three prediction models, including the most accurate ones – XGB for CR and GLM for HS. In Task III, the representation ensemble methods obtained the best results only for HS. In this case, the concatenation method achieved slightly better results than the voting approach, however, the difference in their results was negligible, and they both significantly outperformed the baseline that did not take into account the (dis)similarity between cards (i.e., $\beta = \gamma = 0$). Unexpectedly, for CR the best results were observed for CBOW embeddings, and the second-best were for the proposed *card-end2end* method. Still, the voting ensemble was still better than the considered baseline.

The results of this research have already been applied in our SENSEI framework. Still, we plan some extensions in the near future. For instance, we would like to extend our experiments related to methods of combining representations obtained using different data sources. Although generally, learning a single representation from different data sources seems not always possible due to differences in data granularity and optimization objectives, it would be interesting to consider such an approach for specific cases. It would also be interesting to analyze the effectiveness of the framework for different domains, e.g., different game genres or traditional sports. In particular, we want to verify whether knowledge transfer between embedding learning models can be effectively incorporated. The newest SENSEI's deployment associated with our own aforementioned video game TT is certainly a step towards this direction.

Acknowledgements

This work is a part of the SENSEI project, which was co-financed by EU Smart Growth Operational Programme 2014-2020 under the GameINN project POIR.01.02.00-00-0184/17-00.

References

[1] Akhavian, R., Behzadan, A.H., 2013. Knowledge-based Simulation Modeling of Construction Fleet Operations Using Multimodal-Process Data Mining. Journal of Construction Engineering and Management 139, 04013021. doi:10.1061/(ASCE)CO.1943-7862.0000775.

[2] Arik, S.O., Pfister, T., 2021. TabNet: Attentive Interpretable Tabular Learning, in: Proceedings of AAAI 2021, pp. 6679–6687. URL: https://ojs.aaai.org/index.php/AAAI/article/view/16826.

[3] Bach, F., 2017. Breaking the Curse of Dimensionality with Convex Neural Networks. Journal of Machine Learning Research 18, 1–53. URL: http://jmlr.org/papers/v18/14-546.html.

[4] Bengio, Y., Courville, A., Vincent, P., 2013. Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 1798–1828. doi:10.1109/TPAMI.2013.50.

[5] Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., de Oliveira Pinto, H.P., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., Zhang, S., 2019. Dota 2 with Large Scale Deep Reinforcement Learning. arXiv:1912.06680.

[6] Berrar, D., Lopes, P., Dubitzky, W., 2019. Incorporating Domain Knowledge in Machine Learning for Soccer Outcome Prediction. Machine Learning 108. doi:10.1007/s10994-018-5747-8.

[7] Chen, T., Guestrin, C., 2016. XGBoost: A Scalable Tree Boosting System, in: Proceedings of KDD 2016, pp. 785–794. URL: http://doi.acm.org/10.1145/2939672.2939785, doi:10.1145/2939672.2939785.

[8] Devlin, J., Chang, M., Lee, K., Toutanova, K., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: Proceedings of NAACL-HLT 2019, Volume 1, pp. 4171–4186. doi:10.18653/v1/n19-1423.

[9] Dong, X., Yu, Z., Cao, W., Shi, Y., Ma, Q., 2020. A Survey on Ensemble Learning. Frontiers in Computer Science 14, 241–258. URL: https://doi.org/10.1007/s11704-019-8208-z, doi:10.1007/s11704-019-8208-z.

[10] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, in: Proceedings of ICLR 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[11] Drotár, P., Gazda, M., Vokorokos, L., 2019. Ensemble Feature Selection Using Election Methods and Ranker Clustering. Information Sciences 480, 365–380. doi:https://doi.org/10.1016/j.ins.2018.12.033.

[12] El-Nasr, M.S., Drachen, A., Canossa, A., 2016. Game Analytics. Springer.

[13] García-Sánchez, P., Tonda, A., Squillero, G., Mora, A., Merelo, J.J., 2016. Evolutionary Deckbuilding in Hearthstone, in: Proceedings of IEEE CIG 2016, pp. 1–8.

[14] García, S., Zhang, Z.L., Altalhi, A., Alshomrani, S., Herrera, F., 2018. Dynamic Ensemble Selection for Multi-class Imbalanced Datasets. Information Sciences 445-446, 22–37. doi:https://doi.org/10.1016/j.ins.2018.03.002.

[15] García-Pedrajas, N., Maudes-Raedo, J., García-Osorio, C., Rodríguez-Díez, J.J., 2012. Supervised Subspace Projections for Constructing Ensembles of Classifiers. Information Sciences 193, 1–21. doi:https://doi.org/10.1016/j.ins.2011.06.023.

[16] Huang, F., Zhang, X., Li, Z., Mei, T., He, Y., Zhao, Z., 2017. Learning Social Image Embedding with Deep Multimodal Attention Networks, in: Thematic Workshop Proceedings of ACM Multimedia 2017, pp. 460–468.

[17] Janusz, A., Grad, Ł., Grzegorowski, M., 2019. Clash Royale Challenge: How to Select Training Decks for Win-rate Prediction, in: Proceedings of FedCSIS 2019, pp. 3–6. URL: https://doi.org/10.15439/2019F365, doi:10.15439/2019F365.

[18] Janusz, A., Grad, Ł., Ślęzak, D., 2018a. Utilizing Hybrid Information Sources to Learn Representations of Cards in Collectible Card Video Games, in: Workshop Proceedings of IEEE ICDM 2018, pp. 422–429. URL: https://doi.org/10.1109/ICDMW.2018.00069, doi:10.1109/ICDMW.2018.00069.

[19] Janusz, A., Ślęzak, D., Stawicki, S., Stencel, K., 2018b. SENSEI: An Intelligent Advisory System for the eSport Community and Casual Players, in: Proceedings of IEEE/WIC/ACM WI 2018, pp. 754–757. doi:10.1109/WI.2018.00010.

[20] Janusz, A., Tajmajer, T., Świechowski, M., Grad, Ł., Puczniewski, J., Ślęzak, D., 2018c. Toward an Intelligent HS Deck Advisor: Lessons Learned from AAIA'18 Data Mining Competition, in: Proceedings of FedCSIS 2018, pp. 189–192.

[21] Jeong, Y., Choi, H., Kim, B., Gwon, Y., 2020. DefogGAN: Predicting Hidden Information in the StarCraft Fog of War with Generative Adversarial Nets, in: Proceedings of AAAI 2020, pp. 4296–4303.

[22] Jiang, F., Yu, X., Du, J., Gong, D., Zhang, Y., Peng, Y., 2021. Ensemble Learning Based on Approximate Reducts and Bootstrap Sampling. Information Sciences 547, 797–813. doi:https://doi.org/10.1016/j.ins.2020.08.069.

[23] Justesen, N., Risi, S., 2017. Learning Macromanagement in Starcraft From Replays Using Deep Learning, in: Proceedings of IEEE CIG 2017, pp. 162–169.

[24] Karttunen, J., Kanervisto, A., Kyrki, V., Hautamäki, V., 2020. From Video Game to Real Robot: The Transfer Between Action Spaces, in: Proceedings of ICASSP 2020, pp. 3567–3571. doi:10.1109/ICASSP40776.2020.9053221.

[25] Khalid, S., Khalil, T., Nasreen, S., 2014. A Survey of Feature Selection and Feature Extraction Techniques in Machine Learning, in: Proceedings of SAI 2014, pp. 372–378. doi:10.1109/SAI.2014.6918213.

[26] Khurana, U., Samulowitz, H., Turaga, D., 2018. Feature Engineering for Predictive Modeling Using Reinforcement Learning, in: Proceedings of AAAI 2018 / IAAI 2018 / EAAI 2018.

[27] Lample, G., Chaplot, D.S., 2017. Playing FPS Games with Deep Reinforcement Learning, in: Proceedings of AAAI 2017, pp. 2140–2146.

[28] Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., Teh, Y.W., 2019. Set Transformer: A Framework for Attention-based Permutation-invariant Neural Networks, in: Proceedings of ICML 2019, pp. 3744–3753.

[29] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R.P., Tang, J., Liu, H., 2018. Feature selection: A data perspective. ACM Comput. Surv. 50, 94:1–94:45. URL: `https://doi.org/10.1145/3136625`, doi:`10.1145/3136625`.

[30] Li, W.H., Zhong, Z., Zheng, W.S., 2019. One-pass Person Re-identification by Sketch Online Discriminant Analysis. Pattern Recognition 93, 237–250.

[31] Lu, J., Yang, J., Batra, D., Parikh, D., 2016. Hierarchical Question-image Co-attention for Visual Question Answering, in: Proceedings of NIPS 2016, pp. 289–297.

[32] Luong, T., Pham, H., Manning, C.D., 2015. Effective Approaches to Attention-based Neural Machine Translation, in: Proceedings of EMNLP 2015, pp. 1412–1421. doi:`10.18653/v1/d15-1166`.

[33] Ma, Z., Collins, M., 2018. Noise Contrastive Estimation and Negative Sampling for Conditional Models: Consistency and Statistical Efficiency, in: Proceedings of EMNLP 2018, pp. 3698–3707. doi:`10.18653/v1/d18-1405`.

[34] Matraszek, M., Janusz, A., Świechowski, M., Ślęzak, D., 2021. Predicting Victories in Video Games – IEEE BigData 2021 Cup Report, in: Proceedings of IEEE Big Data 2021, pp. 5664–5671. doi:`10.1109/BigData52589.2021.9671650`.

[35] Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013a. Efficient Estimation of Word Representations in Vector Space, in: Workshop Track Proceedings of ICLR 2013. URL: `http://arxiv.org/abs/1301.3781`.

[36] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J., 2013b. Distributed Representations of Words and Phrases and Their Compositionality, in: Proceedings of NIPS 2013, Volume 2, pp. 3111–3119. URL: `http://dl.acm.org/citation.cfm?id=2999792.2999959`.

[37] Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E.B., Turaga, D., 2017. Learning Feature Engineering for Classification, in: Proceedings of IJCAI 2017, pp. 2529–2535.

[38] Polikar, R., DePasquale, J., Mohammed, H.S., Brown, G., Kuncheva, L.I., 2010. Learn^{++}.MF: A Random Subspace Approach for the Missing Feature Problem. Pattern Recognition 43, 3817–3832. doi:`10.1016/j.patcog.2010.05.028`.

[39] Rao, A., Xu, L., Xiong, Y., Xu, G., Huang, Q., Zhou, B., Lin, D., 2020. A Local-to-global Approach to Multi-modal Movie Scene Segmentation, in: Proceedings of CVPR 2020, pp. 10146–10155.

[40] Sharma, M., Bilgic, M., 2017. Evidence-based Uncertainty Sampling for Active Learning. Data Mining and Knowledge Discovery 31, 164–202. doi:10.1007/s10618-016-0460-3.

[41] Tenney, I., Xia, P., Chen, B., Wang, A., Poliak, A., McCoy, R.T., Kim, N., Durme, B.V., Bowman, S.R., Das, D., Pavlick, E., 2019. What Do You Learn from Context? Probing for Sentence Structure in Contextualized Word Representations, in: Proceedings of ICLR 2019. URL: https://openreview.net/forum?id=SJzSgnRcKX.

[42] Tsumoto, S., Hirano, S., Kimura, T., Iwata, H., 2021. Mining Clinical Process from Hospital Information System: A Granular Computing Approach. Fundamenta Informaticae 182, 181–218. URL: https://doi.org/10.3233/FI-2021-2070, doi:10.3233/FI-2021-2070.

[43] Valle-Lisboa, J.C., Mizraji, E., 2007. The Uncovering of Hidden Structures by Latent Semantic Analysis. Information Sciences 177, 4122–4147. doi:https://doi.org/10.1016/j.ins.2007.04.007.

[44] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is All You Need, in: Proceedings of NIPS 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[45] Wang, R., Ma, W., Tan, M., Wu, G., Wang, L., Gong, D., Xiong, J., 2021. Preference-inspired Coevolutionary Algorithm with Active Diversity Strategy for Multi-objective Multi-modal Optimization. Information Sciences 546, 1148–1165. doi:https://doi.org/10.1016/j.ins.2020.09.075.

[46] Wu, A., Zheng, W.S., Lai, J.H., 2017. Robust Depth-based Person Reidentification. IEEE Transactions on Image Processing 26, 2588–2603.

[47] Xia, P., Zhang, L., Li, F., 2015. Learning Similarity with Cosine Similarity Ensemble. Information Sciences 307, 39–52. doi:https://doi.org/10.1016/j.ins.2015.02.024.

[48] Xia, Y., Chen, K., Yang, Y., 2021. Multi-label Classification with Weighted Classifier Selection and Stacked Ensemble. Information Sciences 557, 421–442. doi:https://doi.org/10.1016/j.ins.2020.06.017.

[49] Zhao, Y.G., Song, Z., Zheng, F., Shao, L., 2018. Learning a Multiple Kernel Similarity Metric for Kinship Verification. Information Sciences 430-431, 247–260. doi:https://doi.org/10.1016/j.ins.2017.11.048.