Duel-based Deep Learning system for solving IQ tests

Paulina Tomaszewska **Adam Żychowski** **Jacek Mańdziuk**
Warsaw University of Technology, Faculty of Mathematics and Information Science,
Koszykowa 75, 00-662 Warsaw, Poland

Abstract

One of the relevant aspects of Artificial General Intelligence is the ability of machines to demonstrate abstract reasoning skills, for instance, through solving (human) IQ tests. This work presents a new approach to machine IQ tests solving formulated as Raven's Progressive Matrices (RPMs), called Duel-IQ. The proposed solution incorporates the concept of a tournament in which the best answer is chosen based on a set of duels between candidate RPM answers. The three relevant aspects are: (1) low computational and design complexity, (2) proposition of two schemes of pairing up candidate answers for the duels and (3) evaluation of the system on a dataset of shapes other than those used for training. Depending on a particular variant, the system reaches up to 82.8% accuracy on average in RPM tasks with 5 candidate answers and is on par with human performance and superior to other literature approaches of comparable complexity when training and test sets are from the same distribution.

1 INTRODUCTION

The motivation behind the launch of Artificial Intelligence (AI) was the ability to transfer human intellectual capabilities to machines, thus accomplishing the Artificial General Intelligence (AGI) level. While this goal is still ahead of us (nowadays we experience Narrow AI) [Adams et al., 2012], some steps on this path have already been made. The well-known Turing test [Turing, 1950] serves for verification whether it is possible to differentiate between a human being and an AI system based on a conversation.

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

More than half of a century later, the so-called visual Turing test was introduced [Geman et al., 2015]. The term refers to Visual Question Answering (VQA) tasks [Antol et al., 2015] where the system is given a question and an image that it should base its answer on. Another approach to testing the intelligence of AI systems is to use the same method as is applied to people, i.e. the IQ tests, which verify the ability of abstract visual reasoning.

1.1 Contribution

In this work, we propose a novel approach to solving visual IQ tests based on Raven's Progressive Matrices (RPMs), called Duel-IQ. The solution is inspired by a round-robin tournament where one candidate answer's fitness is assessed in comparison to another candidate answer.

The motivation behind the idea of a tournament is the relativism of candidate-answer fitness in IQ tests [Raven and Court, 1998]. To clarify this further, let us provide an intuitive example. It may happen that the two candidate answers fit the relations within the RPMs. However, the correct answer is considered the one with a simpler explanation.

For this reason, using a binary model that given a candidate answer is asked to decide whether or not it is correct, may not be an optimal setting. Another possibility would be a classification approach with as many inputs as the number of candidate answers (K) where the model is supposed to point the correct one.

Following our intuition that it could be easier for the model to choose a correct answer out of two choices instead of K possibilities, we propose an approach located between the two above-mentioned extremes, namely with two inputs (candidate answers) compared directly in a *duel*. A series of such duels within a round-robin tournament may, what we believe and verify in the paper, lead to finding the proper answer.

Furthermore, in a tournament scenario, the risk of an incorrect system decision is alleviated within the scor-

ing module. Similarly to ensembles [Dietterich, 2000], some predictions can be wrong, but as the final decision is a result of the aggregation of many predictions (averaging or max voting), it is often closer to the correct one. This feature offers another advantage over using just one model with the number of inputs (filled-in RPMs) equal to the number of candidate answers. An additional asset of the proposed system is its relatively low complexity.

The focus of the paper is on the input design and on the postprocessing of the network output (scoring module). Regarding the latter, we propose two schemes of pairing up candidate answers for the duels, which is also reflected in the number of classifier outputs. Moreover, we evaluate the system performance in the out-of-distribution scenario with a different set of objects (panels) in the training and testing sets, respectively. Further analysis refers to the effects of using auxiliary training.

A performance comparison of the proposed system with three other methods of similar complexity (number of trainable parameters) from the literature is conducted. The results indicate that using a duel-based approach is a meaningful idea that leads to superior accuracy.

The code of the proposed solution and the code for dataset generation are released at `https://github.com/ptomaszewska/Duel-IQ`.

2 PROBLEM DESCRIPTION

The most popular visual IQ tests are based on Raven's Progressive Matrices (RPMs) [Raven and Court, 1998], initially introduced in [Raven, 1936]. Typically, an RPM consists of 9 panels arranged as a 3x3 matrix. The bottom right panel is empty. The test-taker is supposed to choose the correct answer from the set of candidate answers (please refer to Figure 1 for an example) to fill in the empty panel. The correct answer is the one that fits the relations underlying the RPM's design. The relations can occur row-wise, column-wise or/and diagonally.

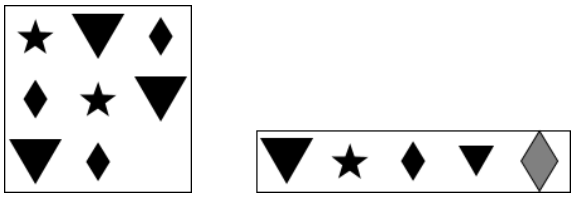

Figure 1: Example of an RPM task (left) with the set of candidate answers (right). The correct answer is the second from the left.

Visual IQ tests are widely used as their results can be compared worldwide. They are independent of language or culture, and require little prior knowledge [Raven, 2000].

3 RELATED WORK

The idea of solving intelligent tasks (i.e. tasks whose solving requires intelligence) by machines is quite old. Examples of such tasks include: detection of the regularities in figures in the geometric-analogy problems [Evans, 1964] (which was the first notable attempt), Bennett's Mechanical Comprehension Tests [Klenk et al., 2011], completion of sequences of numbers [Strannegård et al., 2013] or words-related tests [Ohlsson et al., 2013].

In recent years various Machine Learning (ML) approaches, specifically Deep Learning (DL) methods were proposed to solve RPMs and the related tasks - see [Małkiński and Mańdziuk, 2022a] for a comprehensive overview of this area.

For example, [Hoshen and Werman, 2017] employs convolutional neural networks to recognize the relation between two images and apply this rule to the next image (in one variant by choosing it from a given set of candidates, in another one by generating an answer from a scratch). For the sake of brevity, we will refer to the first variant as "IQ of NN".

DeepIQ [Mańdziuk and Żychowski, 2019] introduces a system with 3 components: a deep autoencoder which is trained to learn a feature-based representation of various figure images, an ensemble of shallow multilayer perceptrons applied to the detection of feature differences, and a scoring module inspired by the human approach used for the final assessment of candidate answers.

The most common approach to solving an RPM relies on identification of the relationships among objects (panels) and using them to identify the correct answer. This idea can be realized by various DL techniques, e.g. Relation Network [Barrett et al., 2018, Zheng et al., 2019], Multiplex Graph Networks [Wang et al., 2020], Scattering Compositional Learner (compositions of a sequence of neural modules) [Wu et al., 2020], or Hierarchical Rule Induction Network [Hu et al., 2021] imitating human induction strategies.

The Wild Relation Network (WReN) introduced in [Barrett et al., 2018], solves an RPM in a few steps. First, 8 RPM panels are concatenated depthwise with all candidate answers (one at a time), resulting in K input packages (K is the number of candidate answer panels). In each package, the panels are embedded us-

ing a convolutional neural network. Then, pairs of such embeddings are input to the relation network whose scores are later aggregated to yield the final output. The above WReN training is extended with an auxiliary training where the network learns to explicitly point the relations, objects and their attributes provided in the input.

Yet another approach [Małkiński and Mańdziuk, 2020] casts the problem of solving RPMs into a multi-label classification framework with labels determined by the set of abstract rules underlying the RPM.

There is no publicly available dataset with the real IQ test questions as they are strictly confidential. Consequently, the authors of the above-mentioned works used artificially generated datasets which vary in complexity, e.g. the number of candidate answers or the selection of geometric transformations applied to the RPM panels. This makes the performance comparison of the solutions proposed in the literature not a straightforward task. In [Chollet, 2019], the benchmark dataset for the assessment of abstract reasoning skills is proposed which, however, does not reflect the structure of IQ tests taken by humans.

A broader look at the field of Abstract Visual Reasoning (of which RPMs are part) is presented in the recent survey paper [Małkiński and Mańdziuk, 2022b].

4 PROPOSED SOLUTION

We propose a duel-based system (see Figure 2) for solving visual IQ tests involving RPMs, called Duel-IQ. The system does not require any expert knowledge and its operation is divided into four steps: preprocessing, feature extraction, classification and scoring.

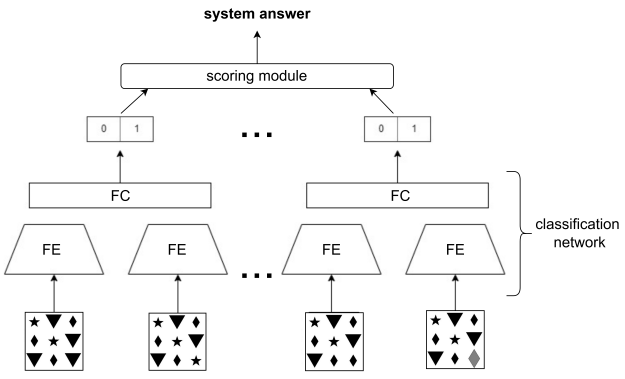

Figure 2: Proposed duel-based system (*winner_in* variant) in which the results of duels are aggregated in the scoring module. FE stands for feature extractor (encoder part of autoencoder), whereas FC for fully connected layers.

During the preprocessing step, RPMs are filled with candidate answers. Therefore, one IQ test task is afterwards represented by as many filled-in RPMs as the number of candidate answers. These RPMs are inputs to subsequent modules. First, the convolutional autoencoder is used for feature extraction (see Figure 2). Then, in the classification network, there are fully connected layers on top of the encoder to judge the winner of the duels (in some scenarios a draw is also possible). The process of pairing up the RPMs filled in with candidate answers and the classifier network are described in more detail in Section 4.2. Predictions of the classifier are later aggregated in the scoring module in order to generate the final decision which of the candidate answers best fits a given RPM (see Section 4.3).

4.1 Autoencoder

The filled-in RPMs are input (one at a time) to the convolutional autoencoder whose task is to recreate the given image. The output of the encoder part provides a compressed representation of the input.

4.2 Classification network

The next step is the dataset preparation for the round-robin duels. Two scenarios are considered at this stage: *winner_in* and *any_in*. In the first one, in the training pairs, there is always a correct candidate answer present. Therefore, the classifier's goal is to choose the correct answer from the two (two-class classification).

In the *any_in* scenario, there is no obligation that there must be a correct answer within the input pair, therefore there may be pairs where none of the candidate answers is correct. In such a case, the model is allowed to choose a third class, meaning a draw - both answers are wrong.

Please observe that at test time (the time of solving an unknown IQ test), the system will experience scenarios where there is no correct answer within the input pair as the pairing up is done in such a way that all combinations are analyzed (irrespective of the order of candidate answers in the pair). Hence, the system trained in the *winner_in* scenario will, at test time, experience the situations it was not trained on. The goal of such a setup was to verify whether the model, in cases where there are two incorrect candidate answers in the input pair, outputs probabilities of the two classes (after the softmax function) of around 0.5 meaning that it is largely uncertain as such a scenario was not previously observed.

The formulated duel pairs are input to the classifi-

cation network. Each RPM is compressed using the encoder and then the two representations are concatenated and passed to dense layers to point a winner. Please recall that the encoder is a part of the trained autoencoder (cf. Section 4.1).

4.3 Scoring module

The results of the duels are aggregated in the scoring module. Two approaches are tested: *probability sum* and *winning frequency*.

4.3.1 *Probability sum scoring*

Let us denote the output of the classifier with softmax in the last layer as $f(x_i, x_j)$ where (x_i, x_j) is the input pair of RPMs filled with candidate-answers $(i \neq j)$. Thus, $f(x_i, x_j)$ is a vector of probabilities where the first element signifies that x_i is the correct answer, and analogically the second element refers to x_j being the correct outcome. In case of the *any_in* scenario, there is the third element in $f(x_i, x_j)$ meaning that both candidate answers are incorrect.

For brevity, we define $f_{x_i}(x_i, x_j)$ as the probability returned by the model specifying that x_i is the correct answer. The probabilities that the answer x_i wins its duels are aggregated in the scoring module:

$$s_{x_i} = \sum_{j \neq i} f_{x_i}(x_i, x_j) \tag{1}$$

Let us denote $s = (s_{x_0}, ..., s_{x_K})$ where K is the number of candidate answers. The final decision of the proposed system is $\hat{y} = \text{argmax}_x\, s$.

4.3.2 *Winning frequency scoring*

Following the notation introduced in the previous section, we formulate the *winning frequency* approach used in the scoring module. Conceptually, it resembles max-voting in ensembles. Here, instead of probabilities, the number of duels wins is considered. Let us denote the equivalent of s_{x_i} from the previous approach:

$$freq_{x_i} = \sum_{j \neq i} I(\underset{x}{\text{argmax}}\, f(x_i, x_j) = x_i) \tag{2}$$

Later, similarly to the *probability sum* approach, the f vector is defined as $f = (freq_{x_0}, ..., freq_{x_K})$. The final answer of the proposed system for a given IQ task, in this scenario, is $\hat{y} = \text{argmax}_x\, f$.

There can happen a particularity when more than one answer has the same number of duels won. In such a case, the system returns the one with a smaller index as the final answer.

5 EXPERIMENTAL SETUP

5.1 Datasets

We use a data generation scheme analogous to that in [Mańdziuk and Żychowski, 2019], but we increase the number of data samples.

The RPM as a whole (i.e. a 3x3 grid of panels) is of the size of 150x150 pixels. Each of the generated test instances reflects a change in one of the following features: shape, size, or rotation. In the case of shape, each row/column contains three different shapes and the task is to point the missing shape in the last row/column. For size and rotation tests, the respective feature (size or rotation) increases or decreases iteratively (with a constant ratio) in consecutive rows/columns. Additionally, in half of the test instances, random perturbations were applied, i.e. apart from the main feature change (described above), another feature was changed randomly (e.g. a figure which iteratively changes its size was randomly rotated).

All figures' features were generated within the following ranges: size (width and height) from 25 to 50 pixels, rotation from 0 to 2π, shading from 0 to 255. Each figure was placed centrally within a panel (a square cell). Please consult [Mańdziuk and Żychowski, 2019] for the details.

To verify the generalization capability of the proposed system, three experiment scenarios were created:

1. the same shapes in training, validation and test datasets (14 figure shapes presented in Figure 3),
2. the figures in training and validation sets were the same (left set of 7 figure shapes in Figure 3) but different than in test set (right set of 7 figure shapes in Figure 3),
3. the same shapes in training, validation and test sets (left set of 7 figure shapes in Figure 3)

Figure 3: Figure shapes used in experiments.

In experiment 2, we wanted to perform a cross-dataset check. To have a baseline to refer to, experiment 3 is run to assess how difficult the task was. In practice, experiment 2 is conducted in such a way that the model is trained on data from experiment 3 but evaluated on testing set from experiment 2. As the number of figure shapes in the training set in experiments 2 and 3 is two times smaller than in the case of experiment 1, we used a two times smaller training dataset (see Table 1).

Table 1: The number of single task questions as shown in Figure 1. In the preprocessing stage, each task is extended to 5 filled-in RPMs that are subsequently combined into pairs for the duels. The training set in experiments 2 and 3 is shared, likewise the validation set. In experiments 2 and 3 the same trained model is tested on two different sets.

	TRAIN	VAL	TEST
experiment 1	34 400	9 800	4 800
experiment 2	17 200	4 900	4 800
experiment 3			4 800

5.2 Autoencoder

The final architecture of the autoencoder (depicted in Figure 4) was chosen by means of a grid search where the number of layers and the numbers of filters in convolutions and transposed convolutions were selected. We analyzed architectures that resulted in different hidden representation sizes and investigated the impact of changing transposed convolution layers into a combination of upsampling and convolution.

It turned out that the proposed architecture (Figure 4) resulted in a lower mean squared error (MSE) and a slightly higher structural similarity index measure (SSIM) than the method with the same latent representation size but with no use of transposed convolution. At the same time, a peak signal-to-noise ratio (PSNR) of our model was slightly lower than that of its fully convolutional counterpart. In conclusion, the proposed architecture presents the best tradeoff between performance metrics (see Section 6) and hidden representation size.

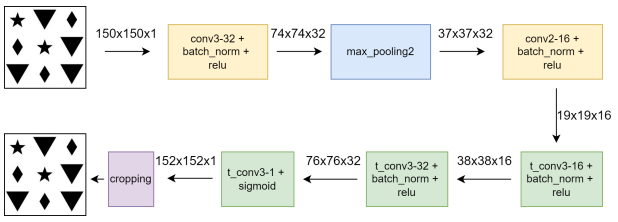

Figure 4: Autoencoder architecture. The convolutional layer parameters are denoted as "conv-kernel size-number of filters". Likewise transposed convolutional layers are described. The max pooling layer parameter refers to the receptive field size.

5.3 Classification

The architecture of the Duel-IQ classifier is shown in Figure 5. The classifier takes a pair of filled-in RPMs that participate in a duel as an input. Both RPMs are compressed using the encoder and then passed through an additional trainable convolutional layer ($kernel\_size = 2, n\_filters = 4$). Next, the two representations are concatenated and processed by two fully connected layers to make the final prediction.

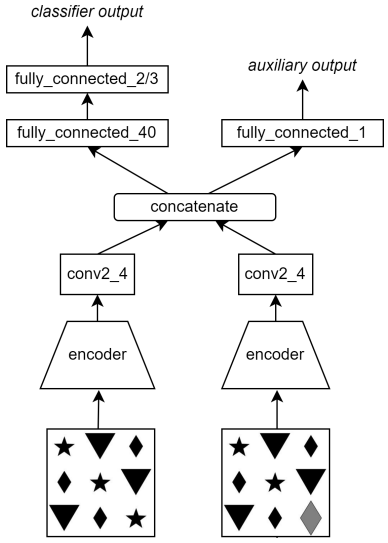

Figure 5: Classification network with auxiliary training applied. Convolutional layers are denoted in the same manner as in Figure 4. A parameter in each fully connected layer refers to the number of neurons.

In order to choose an optimal architecture of the classification network, we checked the impact of making the weights of the encoder trainable as opposed to frozen and observed that enabling fine-tuning is beneficial. We decided to stack an additional convolutional layer on top of the encoder, followed by dense layers, in order to further decrease the size of hidden representations before their concatenation. This convolutional layer is a proxy between the encoder (meant to compress the input in possibly lossless manner) and the dense layers (that focus on comparing the two inputs).

Please note that the networks were not pretrained on Imagenet data (which is a common practice) since RPMs have a different structure than natural scene images. In filled-in RPMs, there are high frequencies and a lot of white space. Moreover, the image is not coherent since it is composed of 9 panels. Furthermore, the images in the dataset are in grayscale and therefore transforming them to RGB space would be redundant and would simply multiply the same information.

In some of the experimental modes, we employed auxiliary training where the model was asked to return the measure of dissimilarity between the candidate answers within the input RPMs. The dissimilarity measure was computed according to Algorithm 1. In the formulas for each term, we applied normaliza-

tion to the [0,1] interval. The most complex is the formula for rotation similarity as it takes into consideration the cyclic nature of the issue. The symbol % means a remainder. The overall dissimilarity measure is also normalized to 0-1, i.e. 0 means exactly the same figure answers, 1 – completely different ones.

Algorithm 1: Dissimilarity measure between two candidate answers.

Input: parameters of candidate answer 1 and 2
$score = 0$
if $shape_1 == shape_2$ **then** $score+ = 1$
$score+ = |intensity_1 - intensity_2|/256$
$score+ = |width_1 - width_2|/50$
$score+ = |height_1 - height_2|/50$
$first\_angle = |rotation_1\%360 - rotation_2\%360|$
$score+ = min(first\_angle, 360 - first\_angle)/180$
Return: $dissimilarity = score/5$

6 EXPERIMENTAL RESULTS

All experiments were run with the following parameters. The training was scheduled for 20 epochs each, however, it could have been stopped earlier if the loss on validation set increased in more than 5 consecutive epochs. The model was optimized using Adam [Kingma and Ba. 2014] with the learning rate 0.001 and decaying coefficients β_1 0.9 and β_2 0.999.

6.1 Autoencoder

The highest spatial compression ratio of the autoencoder with still satisfactory quality of the reconstructed images was 4. The quality was assessed using three metrics: MSE, SSIM and PSNR - see Table 2.

Table 2: Performance metrics of the autoencoder on test data in different experiments (MSE - the lower, the better; SSIM, PSNR - the higher, the better).

	MSE ↓	**SSIM ↑**	**PSNR ↑**
experiment 1	0.002	0.959	29.156
experiment 2	0.003	0.947	27.259
experiment 3	0.001	0.969	30.495

Reconstructed images are of high quality - all structural information is preserved. Note that SSIM is close to the ideal value of 1, whereas PSNR of about 30 is also considered very high.

The best values are achieved in experiment 3 where the task is the easiest because the same figures are used in the training and the testing sets, and only 7 types of object types (shapes) are considered. The second-best scores are in experiment 1 which is similar to experiment 3, but the number of shapes is doubled (there are 14 of them). Experiment 2, in which the model is trained on a dataset containing shapes different from those composing the test set, exhibits the worst performance.

Apart from the verification of quality metrics, a visual inspection was also performed which indicated that despite the drop of SSIM and PSNR measures in experiment 2, the reconstructed images are almost identical to the original images except that the edges of the figures are slightly blurry (cf. Figure 6).

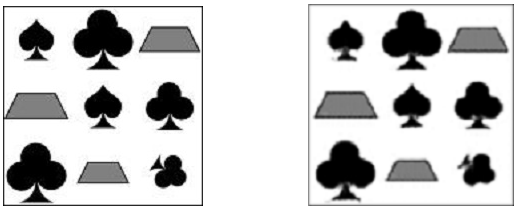

Figure 6: Comparison of original image (left) and the reconstructed one (right).

6.2 Classification network

In total, 8 different variants of classification networks were analyzed. We trained the networks for experiments 1 and 3. In experiment 2, the same model was used as in experiment 3 (training and validation sets are shared in both scenarios). Moreover, we verified whether the choice of the *winner_in* or *any_in* approach has any impact on the performance. Another combination was with and without the auxiliary training applied. Each variant was trained 10 times with different random seeds. Results of these experiments on the test set are presented in Table 3.

The following conclusions can be drawn from the experiments. **(1)** Given the same maximum number of training epochs, there was no significant difference in the results in experiments with and without auxiliary training. **(2)** The metrics were higher in the *winner_in* scenario than in *any_in* which is not that surprising since in the former case a two-class classification was performed, whereas in the latter case, a three-class one. **(3)** The classification network was able to generalize to the test data composed of figures other than those used in the training set (experiment 2). **(4)** The results are visibly better than a random choice that would be 50% in the *winner_in* scenario and 33% in the *any_in*.

6.3 Scoring module

The scoring module was applied on top of the classification variants specified in Table 3. We analyzed

Table 3: Classifier performance on test set in various scenarios - mean accuracy (%) and its standard deviation. For brevity, "exp" stands for experiment.

		INPUT	ACC
with aux. training	exp 1	winner_in	92.7 ± 2.0
		any_in	77.1 ± 7.7
	exp 2	winner_in	70.9 ± 1.2
		any_in	53.1 ± 1.6
	exp 3	winner_in	90.8 ± 0.6
		any_in	83.6 ± 0.8
without aux. training	exp 1	winner_in	92.2 ± 2.5
		any_in	70.0 ± 15.8
	exp 2	winner_in	70.6 ± 1.8
		any_in	53.3 ± 1.3
	exp 3	winner_in	90.3 ± 1.6
		any_in	83.3 ± 1.6

the performance achieved with the two scoring approaches: *probability sum* and *winning frequency* described in Section 4.3. The outcomes are presented in Table 4.

Table 4: Mean test accuracy (%) and its standard deviation of Duel-IQ in various scenarios. The best performance (in terms of mean) in corresponding experiments is presented in bold. "PROB." stands for *probability sum* and "WINNER" for *winning frequency* algorithm in the scoring module.

		INPUT	PROB.	WINNER
with aux. training	exp 1	winner_in	$\mathbf{82.8 \pm 6.2}$	59.1 ± 7.8
		any_in	79.4 ± 7.3	78.4 ± 7.4
	exp 2	winner_in	47.4 ± 1.9	36.4 ± 2.3
		any_in	$\mathbf{49.1 \pm 1.8}$	47.5 ± 1.3
	exp 3	winner_in	77.6 ± 6.6	59.8 ± 4.8
		any_in	73.0 ± 9.1	70.3 ± 10.0
without aux. training	exp 1	winner_in	60.1 ± 11.5	42.7 ± 6.5
		any_in	47.2 ± 10.3	47.3 ± 7.6
	exp 2	winner_in	47.3 ± 2.0	37.5 ± 2.0
		any_in	48.9 ± 1.9	46.8 ± 1.5
	exp 3	winner_in	$\mathbf{80.0 \pm 3.2}$	58.7 ± 5.9
		any_in	77.7 ± 8.0	75.5 ± 10.2

In most of the cases, the high standard deviation in Table 4 is due to the fact that one of the runs resulted in significantly lower performance, whereas the rest of them featured similarly high quality metrics.

The results show that in the *probability sum* scoring, the impact of the auxiliary training is negligible only in the case of experiment 2. In experiment 1, the auxiliary training improved the accuracy, whereas in the case of experiment 3, it slightly deteriorated the performance (however, the difference is within the standard deviation limits). This can be attributed to the fact that in experiment 1, the dataset was twice as big as in experiments 2 and 3.

The overall performance of the system with *probability sum* scoring in *any_in* and *winner_in* scenarios was almost the same in every analyzed case. Note that the classifier accuracy (Table 3) was higher in *winner_in* than in *any_in* scenario. A possible explanation why the advantage in terms of classifier accuracy was not directly translated to the accuracy of the overall system, can be attributed to the fact that the *winner_in* approach has a flavor of out-of-distribution, which makes the problem to be solved more difficult. Therefore, the classifier accuracy in *winner_in* and *any_in* did not transfer in the same way to the overall system's results in Table 4.

A comparison of the *probability sum* and *winning frequency* shows that the results are very similar in the *any_in* scenario whereas significantly different in the *winner_in* scenario where *probability sum* outperforms *winning frequency* approach. This difference can be attributed to the fact that in the *any_in* scenario, the model is not explicitly taught which probabilities it should generate. Therefore, if in the duel with two incorrect answers it favors one of the answers even a little bit, the final model decision in the *winning frequency* approach can be distorted.

6.4 Baseline comparison

We tested our hypothesis and the main motivation behind the proposed system saying that (in this task) it is more effective to perform a series of 2- or 3- class classifications (duels) than a single K-class classification, where K is the number of candidate answers. The latter approach was mostly investigated in previous works. Therefore, we analyzed the analogous neural network architecture to the one from Figure 5 but in a 5-class classification scenario, this time with 5 filled-in RPMs as input. In this case, neither an auxiliary training nor the scoring module were incorporated. To adjust to the increased overall input dimensionality, we changed the number of neurons in the penultimate fully connected layer to 400. The other training parameters were the same as in the baseline Duel-IQ system.

We compared the performance of Duel-IQ with other methods of similar complexity: DeepIQ proposed in [Mańdziuk and Żychowski, 2019], Wild Relation Network (WReN) from [Barrett et al., 2018], and "IQ of NN" from [Hoshen and Werman, 2017].

In the case of WReN, we adjusted the publicly avail-

able code[1] to our dataset features. We also resigned from the auxiliary training due to its fundamentally different nature in WReN [Barrett et al., 2018] compared to our approach. While in Duel-IQ only the dissimilarity measure between the candidate answers is presented during training, in [Barrett et al., 2018] an information about the underlying relations in RPM is provided. The use of this key information makes the overall task of choosing the correct answer much easier for WReN, rendering a direct comparison unfair.

Regarding the "IQ of NN" approach, recall that in the original implementation [Hoshen and Werman, 2017] only the relationship in one row (not the entire RPM) was analyzed. Therefore, we first of all changed the number of input panels. Furthermore, we implemented and tested two versions of this method which are direct extensions of the solution proposed in [Hoshen and Werman, 2017] and differ only in the number of output classes (output neurons). The first one (*5-class*) has 5 outputs (a traditional classification approach), whereas the second one (*2-class*) has only two outputs (a duel-based approach). Results of a series of such duels are further aggregated in the scoring module using the *probability sum* algorithm from Duel-IQ. This option allows verifying the potential benefits of the duel-based approach vs traditional K-class classification in "IQ of NN" model. *winner_in* scenario was investigated as it brought better results in the case of Duel-IQ.

The above-mentioned baseline methods were run 10 times. The accuracy (mean and standard deviation) is presented in Table 5. We summarize the results of Duel-IQ by indicating the highest performance across 8 different variants of the experiment (two pairing schemes: *winner_in*/*any_in*, two algorithms in the scoring module: *probability sum*/*winning frequency*, two training schemes: with/without auxiliary training).

The results prove that Duel-IQ outperforms other analyzed methods when the training and testing datasets consist of the same figure shapes (experiments 1 and 3). Otherwise (experiment 2), the leading method is DeepIQ.

A possible reason for the better performance of DeepIQ in this setting is that it learns the relations between two panels and not the whole filled-in RPM. Such an approach (focusing on two shapes) may lead to developing a more universal representations that help solving an out-of-distribution scenario. Furthermore, DeepIQ is explicitly enforced to focus on predefined aspects such as rotations, changes in size, etc.

However, the drawback of the DeepIQ solution is the need to incorporate expert knowledge for the definition of a possible set of relations within the RPMs. On the contrary, the proposed duel-based system is fully autonomous (does not require *a priori* expert information about the aspects that need to be taken into consideration while solving IQ tests).

[1] https://github.com/Fen9/WReN, https://github.com/mikomel/wild-relation-network

Table 5: Performance comparison of different methods for solving IQ tests: mean, standard deviation, and maximum accuracy (in brackets). Best results in each experiment are in bold.

ALGORITHM	EXP 1	EXP 2	EXP 3
Duel-IQ	$\mathbf{82.8 \pm 6.2}$ (**88.8**)	49.1 ± 1.8 (54.9)	$\mathbf{80.0 \pm 3.2}$ (**85.3**)
Duel-IQ (5-class version)	72.2 ± 5.4 (75.6)	49.2 ± 1.1 (50.8)	71.5 ± 3.2 (74.1)
DeepIQ	73.0 ± 3.7 (78.3)	$\mathbf{61.7 \pm 3.9}$ (**67.4**)	68.8 ± 2.4 (72.5)
WReN	54.1 ± 1.1 (56.3)	34.1 ± 1.2 (36.0)	49.0 ± 1.5 (51.1)
IQ of NN (2-class version)	67.3 ± 20.1 (90.3)	49.1 ± 10.2 (72.8)	68.3 ± 11.2 (89.8)
IQ of NN (5-class version)	55.8 ± 11.5 (76.3)	44.3 ± 5.7 (54.8)	55.1 ± 6.7 (69.3)

Moreover, the experiments confirmed that Duel-IQ results in better accuracy than 5-class classification model of similar architecture (referred to as "Duel-IQ (5-class version)" in Table 5). This observation aligns with the initial hypothesis that classification with two input RPMs can be easier than the one with as many input filled-in RPMs as the number of candidate answers.

Similar conclusions can be drawn from an analysis of the results of the two variants of "IQ of NN". Here, the 2-class version outperforms the 5-class one, as well. However, due to the high variance of "IQ of NN" experimental results, the claim that round-robin approach is beneficial in this case should be taken with care.

The results achieved by "IQ of NN" are mediocre. The possible reason is that the authors used their algorithm for a simplified task with 2 panels and the system was supposed to choose the candidate answer that fits the relation – the third missing panel. This task is much less complex than the analyzed RPMs where there are three rows (matrix) and the relations can occur vertically, horizontally, and/or diagonally. Another aspect is the tendency of the "IQ of NN" to overfit. The authors of the model reported its training for 30 to 100 epochs depending on a problem. When applying the algorithm to our dataset, the model was stopped after 8 to 15 epochs as a result of an early stopping mechanism – the loss on validation set with patience parameter set to 5. The 2-class version achieved the

highest accuracy (in a single run), though, at the same time, the highest variance was reported for this model.

The worst results were achieved with WReN. If figure shapes in the training and testing sets were from the same distribution, the performance of WReN was similar to "IQ of NN (5-class)", otherwise (experiment 2) it was clearly lower.

Quite surprisingly, WReN results were worse in experiment 3 than in experiment 1. This is unexpected because experiment 1 is more difficult than experiment 3 due to higher number of figure shapes. The performance degradation in experiment 3 can be attributed to the fact that in order to train the relation network within WReN, a large number of samples is required (note that the number of training samples in experiment 1 was twice as big as in experiment 3).

In order to assert that the results of Duel-IQ are not biased by the figure shapes chosen for training and testing (see Figure 3), additional tests were performed with randomly shuffled figure shapes between these two sets. The averaged accuracy (over 10 independent runs) in experiments 2 and 3 equaled $51\% \pm 3\%$ and $84\% \pm 3\%$, respectively. The results were generally 2-3% higher compared to the baseline cases (cf. Table 5) which confirms that neither the choice of figures nor their particular split into training and test sets were biased. In fact, the results suggest that the particular setting in the baseline experiments can be considered "more difficult than average".

7 HUMAN PERFORMANCE

The performance of Duel-IQ was confronted with human performance in two surveys conducted among 94 university students and PhD candidates, mostly with Computer Science (CS) background. 61 people took part in the first survey and 33 others in the second one. Each survey was composed of 21 representative RPM problems from our dataset of experiment 1 (7 problem types x 3 instances). The questions (42 in total) varied in their level of difficulty.

The average human accuracy weighted by the ratio of participants in each survey equaled $77.3\% \pm 5.6\%$ which is on par (within standard deviation ranges) with Duel-IQ in experiment 1 ($82.8\% \pm 6.2\%$). Human performance on Raven's Matrices of similar complexity was tested within the general population in [Matzen et al., 2010] with 60% to 80% of correct answers depending on the test type, which is consistent with our survey, taking into account a particular education profile (CS) and age of our participants.

8 CONCLUSIONS

In this paper, we propose a duel-based neural network system (Duel-IQ) suitable for solving RPMs - a well-known example of abstract visual reasoning tasks. The following variants of the system are investigated: (1) the same vs different sets of figures in RPMs in training and test sets, respectively; (2) with vs without auxiliary training; (3) *winner_in* vs *any_in* pairing schemes; (4) *probability sum* vs *winning frequency* scoring formulas.

The results prove that an auxiliary training that enforces the system to assess the similarity between the candidate answers may improve the performance, mainly in the case of the most complex dataset (experiment 1). For *probability sum* scoring function, when the training and test data have the same distribution (experiments 1 and 3), the *winner_in* slightly outperforms the *any_in* approach, whereas in out-of-distribution scenario (experiment 2), it is the other way around. Further investigation of this observation is a focus of our further studies.

Overall, the results confirm the abstract reasoning abilities of the ensemble-like duel-based model and its superiority over 5 reference methods when training and test sets are composed of the same figure shapes. In out-of-distribution setting Duel-IQ is inferior to DeepIQ [Mańdziuk and Żychowski, 2019].

Moreover, the performance of Duel-IQ is on par with the results of a group of 94 CS students and PhD candidates.

Societal impact

Apart from the technical aspects and the properties of the proposed solution, the societal impact of the potential misuse of the system was analyzed. It could hypothetically happen that somebody would use the proposed solution to cheat during online IQ tests and would claim to have a higher IQ than it really is. On a general note, such a fraudulent situation may lead to unequal position in job recruitment process or in other competitive situations referring to IQ scores.

Acknowledgement

The project was funded by POB Research Centre Cybersecurity and Data Science of Warsaw University of Technology within the Excellence Initiative Program - Research University (ID-UB).

References

[Adams et al., 2012] Adams, S., Arel, I., Bach, J., Coop, R., Furlan, R., Goertzel, B., Hall, J., Sam-

sonovich, A., Scheutz, M., Schlesinger, M., Shapiro, S., and Sowa, J. (2012). Mapping the Landscape of Human-Level Artificial General Intelligence. *AI Magazine*, 33:25–42.

[Antol et al., 2015] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. (2015). VQA: Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

[Barrett et al., 2018] Barrett, D., Hill, F., Santoro, A., Morcos, A., and Lillicrap, T. (2018). Measuring abstract reasoning in neural networks. In *ICML*, pages 511–520. PMLR.

[Chollet, 2019] Chollet, F. (2019). On the Measure of Intelligence. *arXiv preprint arXiv:1911.01547*.

[Dietterich, 2000] Dietterich, T. G. (2000). Ensemble Methods in Machine Learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, page 1–15, Berlin, Heidelberg. Springer-Verlag.

[Evans, 1964] Evans, T. G. (1964). A heuristic program to solve geometric-analogy problems. In *Proceedings of the spring joint computer conference*, pages 327–338.

[Geman et al., 2015] Geman, D., Geman, S., Hallonquist, N., and Younes, L. (2015). Visual Turing test for computer vision systems. *Proceedings of the National Academy of Sciences of the United States of America*, 112.

[Hoshen and Werman, 2017] Hoshen, D. and Werman, M. (2017). IQ of neural networks. *arXiv:1710.01692*.

[Hu et al., 2021] Hu, S., Ma, Y., Liu, X., Wei, Y., and Bai, S. (2021). Stratified rule-aware network for abstract visual reasoning. In *AAAI*.

[Klenk et al., 2011] Klenk, M., Forbus, K., Tomai, E., and Kim, H. (2011). Using analogical model formulation with sketches to solve Bennett Mechanical Comprehension Test problems. *Journal of Experimental & Theoretical Artificial Intelligence*, 23(3):299–327.

[Małkiński and Mańdziuk, 2020] Małkiński, M. and Mańdziuk, J. (2020). Multi-Label Contrastive Learning for Abstract Visual Reasoning. *arXiv preprint arXiv:2012.01944*.

[Małkiński and Mańdziuk, 2022a] Małkiński, M. and Mańdziuk, J. (2022a). Deep learning methods for abstract visual reasoning: A survey on Raven's Progressive Matrices. *arXiv preprint arXiv:2201.12382*.

[Małkiński and Mańdziuk, 2022b] Małkiński, M. and Mańdziuk, J. (2022b). A review of emerging research directions in Abstract Visual Reasoning. *arXiv preprint arXiv:2202.10284*.

[Mańdziuk and Żychowski, 2019] Mańdziuk, J. and Żychowski, A. (2019). DeepIQ: A human-inspired AI System for Solving IQ Test Problems. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

[Matzen et al., 2010] Matzen, L. E., Benz, Z. O., Dixon, K. R., Posey, J., Kroger, J. K., and Speed, A. E. (2010). Recreating Raven's: Software for systematically generating large numbers of Raven-like matrix problems with normed properties. *Behavior research methods*, 42(2):525–541.

[Ohlsson et al., 2013] Ohlsson, S., Sloan, R. H., Turán, G., and Urasky, A. (2013). Verbal IQ of a four-year old achieved by an AI system. *Age*, 4(5):6.

[Raven, 2000] Raven, J. (2000). The Raven's progressive matrices: change and stability over culture and time. *Cognitive psychology*, 41(1):1–48.

[Raven, 1936] Raven, J. C. (1936). Mental Tests Used in Genetic Studies: The Performances of Related Individuals in Tests Mainly Educative and Mainly Reproductive. Master's thesis, University of London.

[Raven and Court, 1998] Raven, J. C. and Court, J. H. (1998). *Raven's progressive matrices and vocabulary scales*. Oxford Psychologists Press Oxford, UK.

[Strannegård et al., 2013] Strannegård, C., Amirghasemi, M., and Ulfsbäcker, S. (2013). An anthropomorphic method for number sequence problems. *Cognitive Systems Research*, 22:27–34.

[Turing, 1950] Turing, A. M. (1950). I.—Computing Machinery and Intelligence. *Mind*, LIX(236):433–460.

[Wang et al., 2020] Wang, D., Jamnik, M., and Lio, P. (2020). Abstract diagrammatic reasoning with multiplex graph networks. *arXiv preprint arXiv:2006.11197*.

[Wu et al., 2020] Wu, Y., Dong, H., Grosse, R., and Ba, J. (2020). The scattering compositional learner: Discovering objects, attributes, relationships in analogical reasoning. *arXiv preprint arXiv:2007.04212*.

[Zheng et al., 2019] Zheng, K., Zha, Z.-J., and Wei, W. (2019). Abstract reasoning with distracting features. *arXiv preprint arXiv:1912.00569*.