Predikce nápověd na trase šifrovací hry

Dokumentace projektu

Bára Eliášová

500354

Fakulta informatiky
Masarykova univerzita

7. července 2022

Úvod

Cílem projektu je predikovat, zda si tým vezme nápovědu během hraní šifrovací hry. K této predikci jsou využita data z průchodů týmů hrou a obrazová zadání šifer.

Základní model tvoří neuronová síť, která kombinuje informace o týmu, šifře a celé hře. Tu doplňuje druhá síť, která klasifikuje obtížnost obrazového zadání šifry.

Data

Největší část dat pochází z her pro veřejnost společnosti Cryptomania. Tyto hry jsou řízeny online herním systémem, který tým celou hrou provází a zároveň zaznamenává jeho průchod. Z tohoto záznamu dostal model informace o týmu a o hře.

Obrazová zadání lze rozdělit na dva základní zdroje: šifry od Cryptomania a volně dostupná zadání z veřejných šifrovacích her (přesné počty a rozdělení je rozepsáno v tabulce 1). Data od Cryptomania jsou chráněna autorským zákonem
a vzhledem k tomu, že jsou hlavním artiklem firmy, je nelze najít volně dostupná a nesmí se šířit. Nelze je tedy předat spolu s projektem. Tato data se skládají ze zadání her pro veřejnost
a z šifer, které Cryptomania používá na firemní kurzy.

Hry pro veřejnost jsou vždy motivovány příběhem. Šifry ho dokreslují, proto neobsahují jen samotné zadání, ale často i prvky doplňující atmosféru hry (např. šifra leží na stole, viz obr. 5).

Oproti tomu šifry na firemní kurzy mají velmi jednoduchý styl a kromě samotné šifry obsahují pouze logo firmy, název šifry a copyright v patičce (viz obr. 6). To bylo odstraněno v preprocesingu dat, zbylo tedy čistě zadání úkolu.

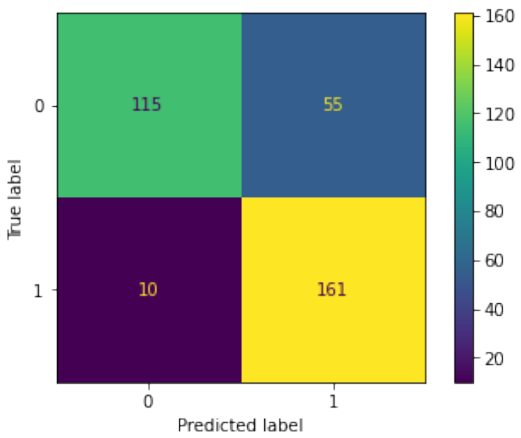
Volně dostupná data byla převzata z archivu šifrovací hry DNEM [1]. Tato hra je neobvyklá tím, že v rámci jednoho ročníku vytvoří tři varianty každé šifry – pro děti, dospělé a experty. Tyto kategorie mají odlišnou složitost, nicméně základ šifry je ve většině případů stejný. Tím lze získat porovnání jak vypadá lehká (dětská) a těžká (pro experty) varianta téže šifry. Všechny šifry obsahují logo a název kategorie obtížnosti, obojí bylo odstaněno při přípravě dat.

Po odstranění loga, názvu, copyrightu a kategorie byly obrázky zmenšeny na čtverec o rozměru 512 na 512 pixelů. Následně byla zadání rozdělena na lehké a těžké šifry. U Cryptomania her pro veřejnost byla využita data z průchodů – pokud nadpoloviční většině týmů trvá vyřešení přes 18 minut, je šifra klasifikována jako těžká (viz tabulka 2). Zbytek šifer byl anotován ručně. Za lehkou josu považovány např šifry na obrázcích 5 a 6, oproti tomu těžká šifra je například na obázku 7.

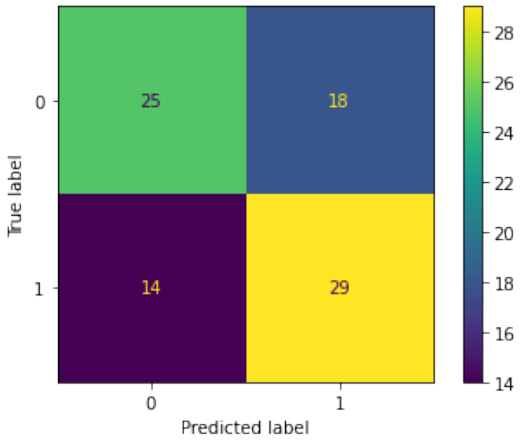
Model pro klasifikaci obtížnosti šifer

Model je tvořen konvoluční neuronovou sítí, která byla předem natrénována na obrazových zadáních šifer. Váhy nejlepšího modelu mohou být načteny pomocí metody `load_weights()` z adresáře `./checkpoints/image_weights` a použity pro klasifikaci nových dat. Model s těmito vahami dosáhl na validační sadě accuracy 0.62, je tedy o něco lepší, než náhodný. S ohledem na malé množství dat je však pochopitelný. Obrázky 1 a 2 ukazují confusion matrices pro trénovací a validační sadu.

[1] `https://www.chameleonbrno.org/dnem/`



Obrázek 1: Confusion matrix pro trénovací sadu



Obrázek 2: Confusion matrix pro validační sadu

Definice třídy modelu

```
class ImageModel:
    def __init__(self, class_names, img_size):
        self.img_height = img_size
        self.img_width = img_size
        self.classes=class_names
        self.num_classes = len(class_names)

        self.model = Sequential([
          layers.Rescaling(1./255, input_shape=(self.img_height, self.
             img_width, 3)),
          layers.Conv2D(16, 3, padding='same', activation='relu'),
          layers.MaxPooling2D(),
          layers.Conv2D(32, 3, padding='same', activation='relu'),
          layers.MaxPooling2D(),
          layers.Conv2D(64, 3, padding='same', activation='relu'),
          layers.MaxPooling2D(),
          layers.Flatten(),
          layers.Dense(128, activation='relu'),
          layers.Dense(self.num_classes, activation="softmax")
        ])

        self.model.compile(optimizer='adam',
                  loss=tf.keras.losses.SparseCategoricalCrossentropy(),
                  metrics=["accuracy"])
```

Model predikující nápovědu

Tento model je opět tvořen neuronovou sítí. Na vstupu má pro tým t na šifře c tato data:

- **Nápovědy**, které si tým doteď vzal ze všech možných. Tedy $\frac{vzal}{2c}$, protože každá šifra nabízí až dvě nápovědy.
- **Čas**, který tým v průměru potřeboval k vyřešení každé z předchozích šifer.
- **Obtížnost šifry**, tedy *lehká*, nebo *těžká*.
- **Pořadí šifry ve hře a hra:** je možné, že si týmy u některých šifer berou nápověu častěji.
- **Výstup předposlední vrstvy modelu pro klasifikaci obtížnosti šifry** pro zadání šifry c.

Výstupem modelu je binární hodnota zda si tým nápovědu vezme (1), nebo ne (0).

Vzhledem k tomu, že jeden řádek dat je tvořen kombinací týmu a konkrétní šifry, je velmi obtížné dataset vyvážit (momentálně si nápovědu vezme přibližně 69% týmů). Z toho důvodu je do modelu přidán `output_bias`, který se snaží nevyváženost kompenzovat.

Váhy nejlepšího modelu jsou uloženy a mohou být načteny pomocí příkazu `hint_model.load_weights("hint_prediction_weights/cp.ckpt")`.

Definice modelu

```
24 class HintModel:
25     def __init__(self, output_bias=None):
26         if output_bias is not None:
27             output_bias = tf.keras.initializers.Constant(output_bias)
28 
29         team_input=tf.keras.layers.Input(shape=(2,),name='team_stats')
30 
31         label_input=tf.keras.Input(shape=(2,),name='easy_hard')
32 
33         task_no_input=tf.keras.Input(shape=(12,),name='task_number')
34 
35         trail_input=tf.keras.Input(shape=(50,),name='trail_id')
36 
37         layer_input=tf.keras.Input(shape=(128,),name='img_layer')
38 
39         y=layers.Concatenate()([team_input, label_input, task_no_input,
               trail_input, layer_input])
40         y=layers.Dense(128, activation='relu')(y)
41         y=layers.Dense(32, activation='relu')(y)
42         y=layers.Dense(64, activation='relu')(y)
43         y=layers.Dense(128, activation='relu')(y)
44         y=layers.Dense(16, activation='relu')(y)
45 
46         y=layers.Dense(32, activation='relu')(y)
```

```
47          y=layers.Dense(1, activation='sigmoid', bias_initializer=output_
               bias)(y)
48
49          self.model = tf.keras.Model(inputs=[team_input, label_input,
               task_no_input, trail_input, layer_input], outputs = [y])
50          self.model.compile(optimizer='adam',
51                             loss='binary_crossentropy',
52                             metrics=['accuracy'])
```

Metoda `train_model(self, x_train, y_train, x_val, y_val, checkpoint_path)`, během trénování ukládá checkpoint do zadaného `checkpoint_path`. Tento checkpoint lze pak načíst pomocí `load_weights()`. Takto načteme váhy nejlepšího modelu a můžeme je použít.

Použití modelu

Pro využití předtrénovaného modelu je třeba vycházet z `main.ipynb`. Tam si nejdřív načteme předzpracovaná data (`load_datasets`), rozdělíme je na trénovací a testovací množinu a nakonec na testovací sadě vyhodnotíme model, viz kód níže.

Potřebné importy

```
53 import json
54 import numpy as np
55 import os
56 from PIL import Image
57
58 from sklearn.model_selection import train_test_split
59
60 import tensorflow as tf
61 from tensorflow.keras import layers
62 from tensorflow.keras.models import Sequential
63 from keras.models import Model
```

Kód využití modelu

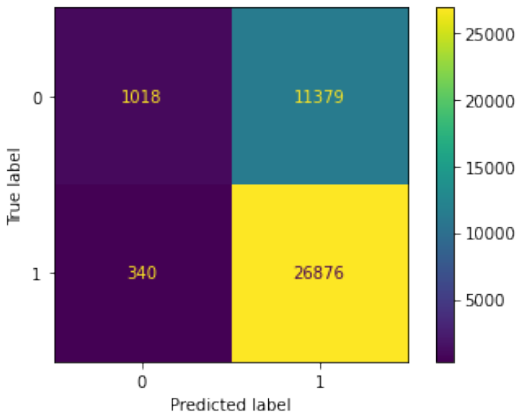
```
64 # main.ipynb
65
66 # nacteni ulozenych dat
67 n_st, n_trl, n_lbl, n_task, n_lay_o, n_h=load_datasets()
68
69 # rozdeleni na trenovaci a testovaci
70 tr_st, val_st, \
71 tr_trl, val_trl, \
72 tr_lbl, val_lbl, \
73 tr_task, val_task, \
74 tr_lay, val_lay, \
```

```
tr_h, val_h = train_test_split(n_st, n_trl, n_lbl, n_task, n_lay_o, n_h,
    train_size=0.9, random_state=42)

# prevedeni dat na tensory
tr_st=tf.convert_to_tensor(tr_st)
val_st=tf.convert_to_tensor(val_st)

tr_trl=tf.one_hot(tr_trl,tr_trl.max()+1)
val_trl=tf.one_hot(val_trl,val_trl.max()+1)

tr_lbl=tf.one_hot(tr_lbl,2)
val_lbl=tf.one_hot(val_lbl,2)

tr_task=tf.one_hot(tr_task,tr_task.max())
val_task=tf.one_hot(val_task,val_task.max())

tr_lay=tf.convert_to_tensor(tr_lay)
val_lay=tf.convert_to_tensor(val_lay)

# kompletace trenovaci a testovaci mnoziny
x_train=[tr_st, tr_lbl, tr_task, tr_trl, tr_lay]
y_train=tr_h
x_val=[val_st, val_lbl, val_task, val_trl, val_lay]
y_val=val_h

# evaluace modelu
my_model=HintModel(None)
my_model.set_checkpoint_path("hint_prediction_weights/cp.ckpt")
my_model.load_weights()
my_model.model.evaluate(x_val, y_val, verbose=2)
```

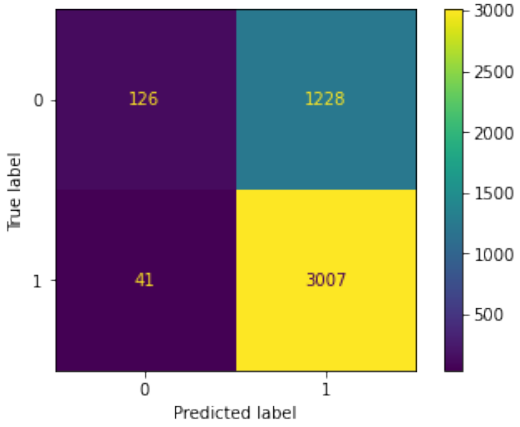
Výsledky a vyhodnocení

Vzhledem k povaze úlohy byla jako rozhodující veličina zvolena accuracy. Protože je dataset nevyvážený, nelze vycházet jen z její hodnoty, ale je třeba ji porovnat s daty, až pak je možné říct, zda se model něco naučil a nevrací např. samé nuly. Z analýzy vyplynulo, že v trénovací sadě je pozitivních příkladů 68.7% a v testovací 69.2%, tyto hodnoty tedy chceme s accuracy překonat. Nejlepší model dosáhl hodnot 70.4% na trénovací sadě a 71.2% na testovací. Lze tedy říci, že je model o něco lepší, než náhodný. Obrázky 3 a 4 ukazují confusion matrices pro trénovací a validační sadu. Model očividně predikuje výrazně více 1, což je pravděpodobně dáno nevyvážeností datasetu.

Jako další informace, které je možné modelu poskytnout se nabízí extrakce textu ze zadání šifry, tento postup se však ukázal nemožným, protože většina šifer text neobsahuje a analýza tohoto textu nepřinesla nic užitečného. Dále je možné přidat legendu, která je k šifře doplněna v online herním systému.



Obrázek 3: Confusion matrix pro trénovací sadu



Obrázek 4: Confusion matrix pro validační sadu

Ukázka dat a statistiky

Vzhledem k nemožnosti přiložit celý dataset, přikládám ukázku dat pro lepší představu. Z tabulky 2 je zřejmé, že nebyly použity všechny šifry, je to dáno tím, že ne všechny mají obrazové zadání, které by mohlo být v analýze použito.

Sada/label	**Zdroj dat a jejich počet**			
	DNEM	**firemní kurzy**	**hry pro veřejnost**	**celkem**
train/easy	79	56	35	170
train/hard	84	52	35	171
valid/easy	21	14	8	43
valid/hard	23	11	9	43
celkem/easy	100	70	43	213
celkem/hard	107	63	44	214

Tabulka 1: Zdroje obrazových dat pro predikci obížnosti.

Tabulka 2: Přehled dat z her pro veřejnost.

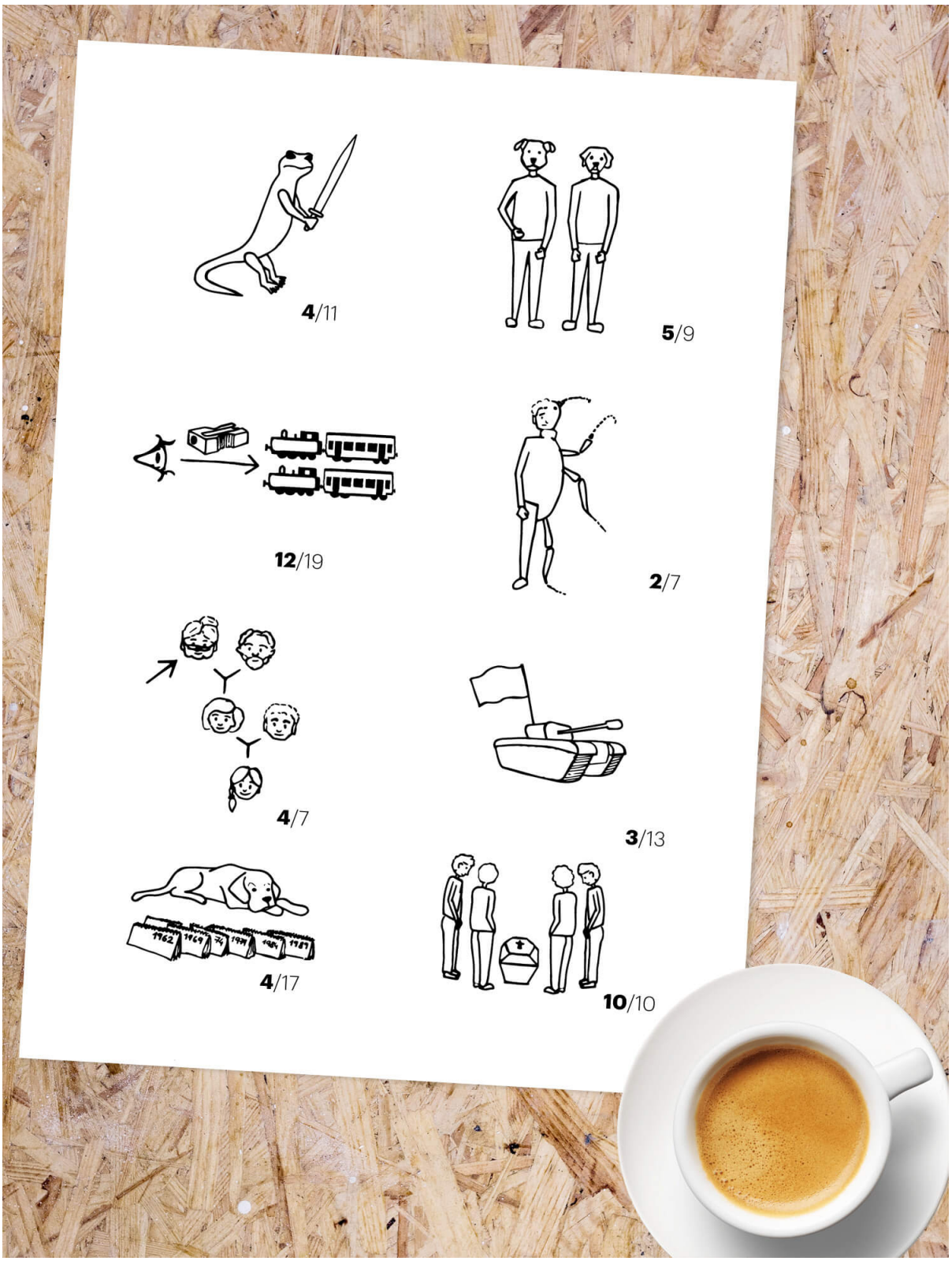
Začátek tabulky			
Pořadí šifry ve hře	**průměrný čas (minuty)**	**label**	**sada**
hra Avraham Harshalom			
1	10.0	easy	train
2	9.2	easy	train
3	14.0	easy	train
5	10.9	easy	val
hra Dopis bez adresy			
1	8.7	easy	train
2	12.9	easy	train
4	20.5	hard	train

Pokračování tabulky 2

Pořadí šifry ve hře	**průměrný čas (minuty)**	**label**	**sada**
5	14.1	easy	train
hra Fantom Brna			
2	4.4	easy	train
3	6.8	easy	train
4	8.5	easy	train
5	19.7	hard	train
6	13.4	easy	train
7	22.3	hard	train
9	37.2	hard	train
10	15.6	easy	train
hra Královské mysterium			
1	13.5	easy	train
2	15.9	easy	train
3	28.2	hard	train
6	17.1	easy	val
hra Loupež po telefonu			
1	18.2	hard	train
2	3.9	easy	train
3	6.9	easy	train
5	44.4	hard	val
8	18.8	hard	train
hra Moravský Manchester			
1	57.5	hard	val
3	10.4	easy	train
4	14.5	easy	train
6	13.9	easy	val
7	13.4	easy	train
8	14.0	easy	train
hra Obrazy Josefa Temperníka			
1	20.8	hard	val
2	7.7	easy	train
3	49.8	hard	train
4	90.1	hard	val
5	112.0	hard	train
6	173.1	hard	train
7	204.5	hard	val
8	12.2	easy	train
hra Osmý div světa			
1	21.6	hard	train
2	30.4	hard	train
3	64.1	hard	train
4	57.4	hard	train

Pokračování tabulky 2			
Pořadí šifry ve hře	**průměrný čas (minuty)**	**label**	**sada**
7	39.6	hard	train
hra Před pikolou, za pikolou...			
1	19.7	hard	train
2	28.1	hard	train
3	48.7	hard	train
7	19.3	hard	val
8	128.6	hard	train
9	58.4	hard	train
10	54.9	hard	train
hra Příběh Enigmy			
1	12.4	easy	val
2	33.9	hard	train
4	61.6	hard	train
5	61.4	hard	train
6	24.9	hard	train
7	23.8	hard	val
8	53.1	hard	train
10	26.0	hard	train
hra Staré pověsti české			
4	16.1	easy	train
5	10.7	easy	train
6	4.9	easy	train
7	2.9	easy	train
8	18.1	hard	val
9	5.1	easy	val
11	5.9	easy	val
12	22.9	hard	train
hra Sedm klíčů			
1	11.3	easy	train
2	27.0	hard	train
3	20.0	hard	train
4	29.5	hard	train
5	45.0	hard	train
6	6.6	easy	train
hra Šeptající javor			
3	9.0	easy	train
4	13.9	easy	train
6	14.4	easy	train
hra Ve stínu černé vrány			
1	6.2	easy	train
4	55.6	hard	train
5	9.2	easy	train
6	5.1	easy	val

Pokračování tabulky 2			
Pořadí šifry ve hře	**průměrný čas (minuty)**	**label**	**sada**
7	34.5	hard	val
8	14.6	easy	train
hra Ztracené židovské město			
1	17.5	easy	train
3	18.1	hard	train
4	10.9	easy	train
5	17.2	easy	val
6	23.9	hard	train
Konec tabulky			



Obrázek 5: Ukázka šifry ze hry pro veřejnost

Obrázek 6: Ukázka šifry z firemního kurzu

Obrázek 7: Ukázka těžké šifry z šifrovací hry DNEM