VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

METODY PRO SHLUKOVÁNÍ A VYHLEDÁVÁNÍ V OBRAZOVÝCH DATECH Z ELEKTRONOVÝCH MIKROSKOPŮ

METHODS FOR CLUSTERING AND SEARCHING IN VISUAL DATA FROM ELECTRON MICROSCOPY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. TOMÁŠ PLACHÝ**
AUTHOR

VEDOUCÍ PRÁCE **doc. Ing. MARTIN ČADÍK, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání diplomové práce

147474

Ústav:	Ústav počítačové grafiky a multimédií (UPGM)
Student:	**Plachý Tomáš, Bc.**
Program:	Informační technologie a umělá inteligence
Specializace:	Inteligentní systémy
Název:	**Metody pro shlukování a vyhledávání v obrazových datech z elektronových mikroskopů**
Kategorie:	Zpracování obrazu
Akademický rok:	2022/23

Zadání:

1. Seznamte se s metodami pro shlukování a vyhledávání v obrazových datech s pomocí i bez pomoci metadat.
2. Vytipujte metody vhodné pro shlukování a vyhledávání obrazových dat z elektronových mikroskopů. Vlastnosti vybraných metod popište.
3. Navrhněte a implementujte systém pro shlukování a vyhledávání obrazových dat z elektronových mikroskopů. Do systému implementujte vybrané metody.
4. S daným systémem experimentujte, porovnejte čistě obrazové metody s metodami založenými na metadatech, vyhodnoťte dosažené výsledky, diskutujte možnosti budoucího vývoje, případně navrhněte vlastní modifikace implementovaných metod.
5. Dosažené výsledky prezentujte formou videa a plakátu nebo článku.

Literatura:

- BENGIO, Yoshua; GOODFELLOW, Ian; COURVILLE, Aaron. *Deep learning*. Cambridge, MA, USA: MIT press, 2017.
- VAN GANSBEKE, Wouter, et al. Scan: Learning to classify images without labels. In: *European conference on computer vision*. Springer, Cham, 2020. p. 268-285.
- NIU, Chuang; SHAN, Hongming; WANG, Ge. Spice: Semantic pseudo-labeling for image clustering. *arXiv preprint arXiv:2103.09382*, 2021.

Při obhajobě semestrální části projektu je požadováno:
Body 1-3 zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce:	**Čadík Martin, doc. Ing., Ph.D.**
Konzultant:	Ing. Lukáš Hübner
Vedoucí ústavu:	Černocký Jan, prof. Dr. Ing.
Datum zadání:	1.11.2022
Termín pro odevzdání:	17.5.2023
Datum schválení:	8.12.2022

Abstrakt

Tato práce řeší problematiku shlukování obrázků z elektronové mikroskopie. Tyto obrázky lze shlukovat podle vizuální podobnosti nebo podle metadat zachycujících nastavení mikroskopu v době pořízení obrázku. Cílem práce je porovnat úspěšnosti obou výše zmíněných přístupů a prozkoumat možnost využití shlukování k rozdělení sady obrázků na obrázky korektní a obrázky zachycující chybu vzniklou při automatizované práci mikroskopu. Hlavním závěrem práce je zjištění, že mezi korektním a chybným obrázkem z elektronového mikroskopu je vizuální rozdíl i rozdíl v metadatech tak malý, že je pomocí shlukování bez učitele nelze správně odlišit. Pozitivním přínosem práce je ale demonstrování použitelnosti vybraných metod na automatické rozdělení sady obrázků do tříd odpovídajících různým fázím práce mikroskopu, což usnadní pracovníkům manuální analýzu obrázků.

Abstract

This thesis deals with the problem of clustering images from electron microscopy. These images can be clustered by visual similarity or by metadata, which describe the settings of the microscope. The goal of this thesis is to compare these two clustering approaches and to explore the possibility of utilizing clustering to split a set of pictures into two parts - one containing correct pictures and the other containing pictures which capture an error during automatized work of an electron microscope. Conclusion of this thesis is that visual differences and differences in metadata between correct and errorous images from electron microscopy are so small, that they cannot be distinguished by unsupervised clustering techniques. However, a positive contribution of this work is demonstration of usability of the methods chosen in this thesis for clustering images into groups corresponding with different phases of work of the microscope, which will make the manual analysis of these pictures easier.

Klíčová slova

shlukování, porovnání shlukování na základě obrazových dat a metadat, elektronová mikroskopie, neuronové sítě, učení bez učitele

Keywords

clustering, comparison of clustering based on image data and metadata, electron microscopy, neural networks, unsupervised learning

Citace

PLACHÝ, Tomáš. *Metody pro shlukování a vyhledávání v obrazových datech z elektronových mikroskopů.* Brno, 2023. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. Ing. Martin Čadík, Ph.D.

Metody pro shlukování a vyhledávání v obrazových datech z elektronových mikroskopů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením doc. Ing. Martina Čadíka, Ph.D. a pod dohledem Ing. Lukáše Hübnera coby externího konzultanta z firmy Thermo Fisher Scientific a že jsem uvedl všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....

Tomáš Plachý

16. května 2023

Poděkování

Děkuji panu docentovi Čadíkovi za vedení a konstruktivní kritiku v průběhu práce. Dále bych rád poděkoval firmě Thermo Fisher Scientific za poskytnutí dat a panu inženýrovi Hübnerovi za důvěru s vypracováním tohoto zadání.

Obsah

Kapitola 1

Úvod

Elektronová mikroskopie v dnešní době proniká do většiny vědních disciplín. Ať už se jedná o zkoumání DNA, vývoj mikročipů či výrobu nano-materiálů, elektronové mikroskopy jsou nedílnou součástí pokroku moderní vědy. Hlavní směr vývoje v elektronové mikroskopii byl dlouhá léta zaměřen na zvyšování rozlišení mikroskopů. S rostoucí náročností pokroku v této oblasti se ale začal vývoj orientovat i směrem k automatizaci práce mikroskopu. Ta se mnohdy sestává ze dvou hlavních kroků - vyříznutí a ztenčení plíšku (nebo-li lamely) studovaného materiálu v dual-beamovém mikroskopu a následného přenesení lamely do jiného typu mikroskopu, který ji prozáří elektronovým svazkem a získá tak nejlepší možné rozlišení.

Tato práce je součástí projektu AutoTEM firmy Thermo Fisher Scientific, jehož cílem je plně automatizovat právě proces výřezu lamely pomocí malého dual-beamového elektronového mikroskopu. V tomto procesu může nastat velké množství různých chyb, které lze pomocí senzorů v mikroskopu odhalit až po dokončení celé jedné části úkolu, což znamená potenciální ztrátu vyšších desítek minut práce mikroskopu.

Možným řešením tohoto problému je rozpoznat chybu za běhu z obrazového výstupu mikroskopu pomocí klasifikačních neuronových sítí. K natrénování těchto sítí jsou však potřeba anotované datasety čítající stovky až tisíce obrázků, což podstatně zvyšuje nákladnost a dobu vývoje řešení.

V této práci prozkoumám možnost automatické tvorby datasetů pro trénování klasifikačních neuronových sítí. K tomu je třeba provést rozdělení obrázků z minulých běhů mikroskopů do potřebných tříd (například na obrázky s chybou a bez chyby ze specifického kroku procesu na elektronovém mikroskopu). Za tímto účelem se seznámím s moderními metodami pro shlukování podle vizuální podobnosti, vyberu metody vhodné pro použití na datasetech z elektronové mikroskopie a vyhodnotím jejich úspěšnost.

Je ovšem třeba prozkoumat i možnost, že je informace o chybě zakódovaná ve specifické kombinaci hodnot nastavení elektronového mikroskopu, které se ukládají coby metadata s pořízeným obrázkem. Proto provedu shlukování i podle metadat pomocí tradičních metod pro shlukování bez učitele. Jedním z hlavních cílů práce je potom porovnat výsledky shlukování podle vizuální podobností se shlukováním podle metadat.

Dalším problémem, se kterým si firma Thermo Fisher Scientific potýká, je potřeba zaměstnanců rozdělit soubor obrázků na vizuálně podobné skupiny za účelem manuální analýzy běhu mikroskopu. Rozdělení lze sice docílit ručním filtrováním na základě metadat, ale nalezení správného parametru, který obrázky rozdělí požadovaným způsobem, vyžaduje hlubší znalost metadat a je tedy časově náročné obzvlášť pro nové zaměstnance. Proto otestuji použitelnost vybraných metod pro shlukování bez učitele i na tento problém.

Pro měření úspěšnosti modelů vytvořím anotované datasety, které budou odpovídat výše popsaným modelovým případům použití shlukování. Pomocí těchto datasetů otestuji a porovnám úspěšnost různých přístupů ke shlukování bez učitele (anotace budou však vždy použity pouze k měření kvality shlukovacích vlastností modelu, ne k jeho trénování) a zhodnotím jejich použitelnost k řešení problémů v praxi.

Kapitola 2

Učení bez učitele a shluková analýza

Modely testované v této diplomové práci spadají způsobem učení do kategorie učení bez dozoru či učení bez učitele [10, 17]. V této kapitole stručně charakterizuji tento typ učení a jeho použití při řešení problému shlukování.

Učení bez učitele je metodou strojového učení, při které nejsou vstupní data provázána s očekávanými výstupy modelu. Učení tedy nespočívá v úpravách vnitřního nastavení modelu na základě porovnání reálného výstupu s výstupem očekávaným, ale ve snaze o vytvoření smysluplné reprezentace vstupních dat, díky které bude možno rozdělit datové objekty do tříd.

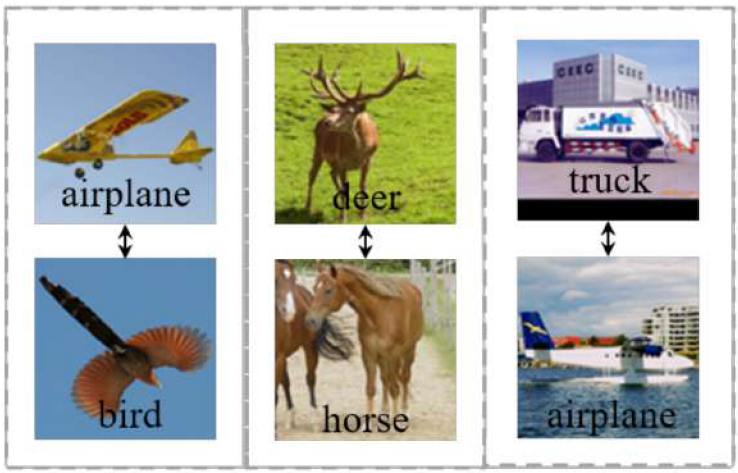
Za tímto účelem jsou objekty umístěny do n-dimenzionálního prostoru, kde n značí počet kategorií informací, které máme o objektu k dispozici (například počet různých atributů v databázi či délka embeddingového[1] vektoru), díky čemuž můžeme podobnost objektů získat vyhodnocením vzdálenostní funkce (například jednoduše vypočtením Euklidovské vzdálenosti) mezi datovými body. Dobrá metoda shlukové analýzy potom v tomto prostoru vytváří shluky s vysokou podobností datových objektů uvnitř tříd a malou podobností objektů mezi třídami.

2.1 Shlukování podle obrazových dat

Při řešení problému shlukování podle obrazových dat (tedy podle hodnot barevných kanálů jednotlivých pixelů vstupních obrázků) je učení často rozděleno do několika fází. V první fázi se model učí reprezentovat vstupní obrázky pomocí vektorů čísel, neboli embeddingů. Učení spočívá ve vygenerování embeddingu pro vstupní obrázek a pro jeho upravenou verzi (mezi úpravy patří například rotace, změna jasu a kontrastu nebo oříznutí obrázku) a následné úpravě vah modelu takovým způsobem, aby model generoval co nejpodobnější reprezentaci těchto dvou obrázků. V nejjednodušším provedení je pak v následujícím kroku na takto získané body v embeddingovém prostoru aplikována metoda K-means clustering (či jiný shlukovací algoritmus založený na výpočtu vzdálenostní funkce [7, 15]), která prostor rozdělí na požadovaný počet shluků.

[1] Embedding - v tomto kontextu výstup po zpracování obrázku neuronovou sítí. Jedná se o reprezentaci obrázku posloupností n číslic, na kterou lze nahlížet jako na souřadnice bodu v n-dimenzionálním prostoru.

Modely pro shlukování obrazových dat popsané v této práci se od sebe liší v krocích, které následují po výše zmíněném učení smysluplné reprezentace. Cílem následujících kroků je často vyřešit závislosti na nízkoúrovňových informacích z obrázku (například na barvách, které nemusejí být důležité pro rozlišení dvou různých, ale podobně barevných objektů - viz 2.1), tento první krok učení zůstává i u SOTA (State Of The Art) modelů stejný [16, 31].



Obrázek 2.1: Vizuálně podobné ale sémanticky odlišné obrázky, které jsou typicky těžké rozlišit při shlukování bez učitele. Převzato z [4].

2.2 Shlukování podle metadat

Shlukování podle metadat lze převést na klasický problém shlukování databáze, ve které každý řádek obsahuje metadata (neboli hodnoty nastavení mikroskopu v době pořízení obrázku) uložená k jednomu specifickému obrázku. Hlavní výzvou je tedy příprava heterogenních dat a výběr vzdálenostní funkce. Pro samotné shlukování se pak použijí klasické shlukovací metody [7, 15].

Metadata obrázků z elektronových mikroskopů obsahují intervalové a kategorické atributy. Před samotnou analýzou je tedy třeba normalizovat intervalová data do intervalu $< 0, 1 >$ a vytvořit kontingenční tabulku neshod kategorických atributů mezi jednotlivými datovými objekty. Výslednou vzdálenost mezi dvěma objekty získáme pomocí Gowerova algoritmu [19], což je metoda navržená pro vypočtení vzdálenosti mezi datovými body, které obsahují jak kategorická, tak intervalová data. Finálním krokem analýzy je pak vypočtení matice Gowerových vzdáleností mezi všemi datovými body a následné shlukování na základě této matice.

Kapitola 3

Modely pro shlukování na základě obrazových dat

V této kapitole stručně shrnu klíčové vlastnosti vybraných modelů pro shlukování bez učitele na základě obrazových dat. Na závěr vyberu kandidátní model, jehož použitelnost otestuji na datasetech obrázků z elektronových mikroskopů firmy Thermo Fisher Scientific (dále jen TFS).

Odhadovaný počet tříd v datasetu obrázků z elektronových mikroskopů je v řádu jednotek až nízkých desítek, proto byly vybrány tři poslední SOTA modely na základě jejich úspěšnosti na testovacím datasetu CIFAR-10 [25].

3.1 SPICE - Semantic Pseudo-Labeling for Image clustering

SPICE [31] je současný SOTA model pro shlukování neanotovaných obrázků, který oproti svým předchůdcům představuje nový algoritmus pro důvěryhodné pseudo-anotování neanotovaných dat a dosahuje úspěšnosti 91.8 % na datasetu CIFAR-10.

3.1.1 Části modelu

SOTA výsledků dosahuje model rozdělením neuronové sítě pro shlukování do dvou částí a to na Feature[1] Model (zkráceně FeaModel) a shlukovací hlavu (neboli samostatnou dvouvrstvou neuronovou síť).

Rozdělením modelu na tyto dvě části bylo dosaženo nejen větší kontroly nad závislostí výstupu na nízkoúrovňových vlastnostech obrázku, ale i snížení výpočetní náročnosti. Díky oddělení shlukovací hlavy, jejíž trénování je výrazně méně náročné, můžeme natrénovat více shlukovacích hlav nezávisle na sobě zároveň a vybrat tu nejlepší, čímž snížíme závislost výsledků na prvotní náhodné inicializaci vah.

Feature model

FeaModel je nejdříve natrénován samostatně pro měření podobnosti obrázků pouze podle vizuálních informací. Pracuje tedy pouze s hodnotami barevných kanálů jednotlivých pixelů, na základě kterých vytváří embeddingový vektor pro reprezentaci těchto obrázků v n-dimenzionálním prostoru.

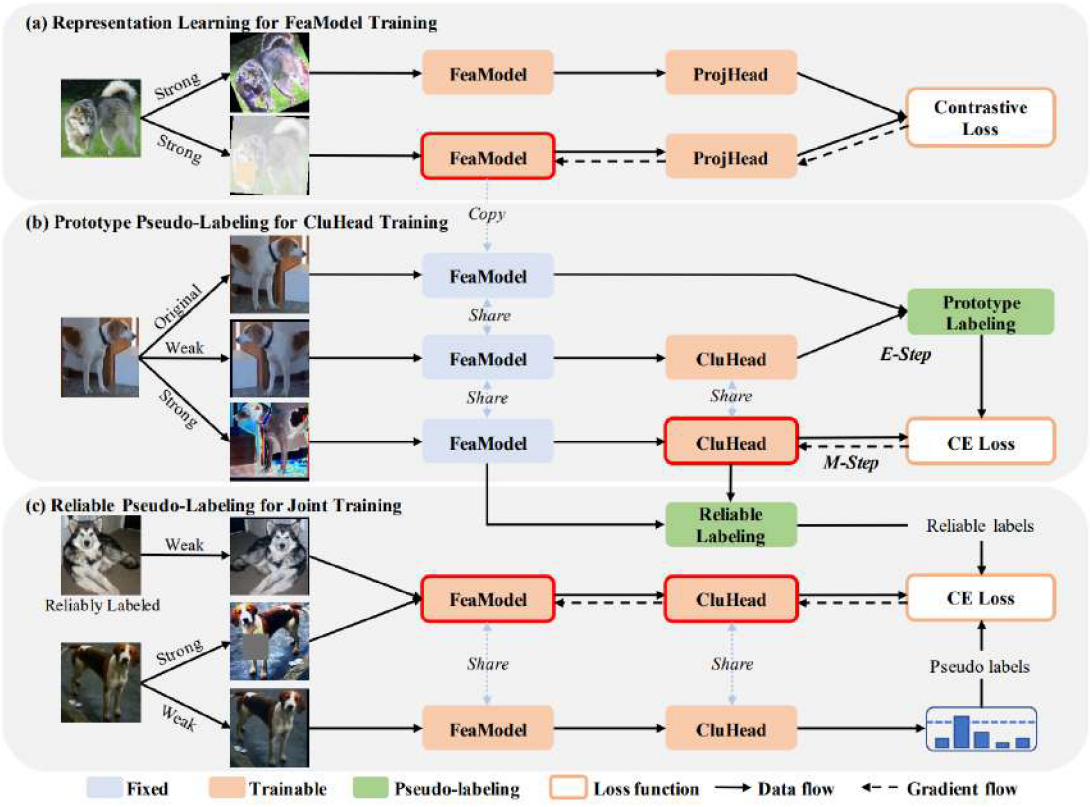
[1] Feature - v tomto kontextu informace o vlastnostech určité oblasti v obrázku (od výskytu hran po výskyt více abstraktních objektů, například obličeje)

Shlukovací hlava

Shlukovací hlava je potom dvouvrstvá neuronová síť trénovaná pro predikování pravděpodobnosti náležitosti vzorku do jednotlivých shluků na základě výstupního vektoru z FeaModelu. K trénování této funkce však nejsou dostupné ground-truth[2] třídy a použijí se pseudo-labely[3] (také nazývané pseudo-třídy) získané pomocí Expectation Maximization algoritmu (viz 3.1.2).

3.1.2 Fáze trénování

Trénování modelu je rozděleno na tři hlavní kroky, jak je znázorněno na obrázku 3.1. FeaModel je nejdříve trénován samostatně a poté je už bez úprav vah použit pro trénování shlukovací hlavy. Na závěr jsou obě části modelu dotrénovány společně.



Obrázek 3.1: Fáze trénování modelu SPICE. Převzato z [31].

[2]Ground-truth v kontextu anotovaných datasetů znamená označení náležitosti vzorku ke třídě odpovídající realitě.

[3]Pseudo-labely jsou označení náležitosti jednotlivých vzorků ke třídě odhadnuté modelem.

Krok 1 - trénování FeaModelu

Nejdříve se trénují paralelně dvě větve s různými FeaModely, které budeme nazývat horní a spodní. Obě obdrží vlastní náhodnou transformaci vstupního obrázku s tím, že cílem trénování v této fázi je, aby výstupní vektory různých transformací jednoho obrázku byly podobné a výstupní vektory dvou různých obrázků různé. V průběhu trénování se ale optimalizují parametry (metodou Contrastive learning) pouze modelu na spodní větvi a horní model je aktualizován coby moving-average parametrů spodního.

Krok 2 - trénování shlukovací hlavy

V této fázi se trénují tři větve se třemi FeaModely (jejichž váhy se v této fázi trénování už neupravují), na kterých jsou ve spodních dvou větvích napojeny shlukovací hlavy.

- Horní - tvoří ji pouze FeaModel. Bere na vstupu neupravený obrázek a vrací embeddingový vektor.
- Prostřední - skládá se z FeaModelu a shlukovací hlavy. Bere na vstupu slabě upravený obrázek a odhaduje pravděpodobnost náležitosti k jednotlivým shlukům. V kombinaci s výstupem horní větve generuje pseudo-labely (Expectation fáze).
- Spodní - skládá se z FeaModelu a shlukovací hlavy. Bere na vstupu silně upravený obrázek a optimalizuje parametry shlukovací hlavy podle pseudo-labelů (Maximization fáze).

Expectation fáze: Shlukovací hlava v prostřední větvi odhadne pravděpodobnosti náležitosti obrázku k jednotlivým shlukům a přiřadí k embedingu obrázku z hodní větve pseudotřídu. Po zpracování celé mini-batch obrázků dojde k výpočtu centra shluku v embedingovém prostoru každé pseudo třídy a ke každému centru je poté přiřazeno $\frac{K}{M}$ nejbližších obrázků, kde M je počet shluků a K počet obrázků v mini-batchi.

Maximization fáze: Na závěr celého kroku se použije silně upravená ale již anotovaná mini-batch obrázků z předchozí Expectation fáze k natrénování shlukovací hlavy ve spodní větvi učením s učitelem. Jelikož je ale tato pseudo-anotace méně spolehlivá, než trénování se znalostí opravdových ground-truth tříd, požívá se k získání výstupního vektoru dvakrát za sebou softmax funkce. Následkem je zpomalení učící rychlosti obzvláště u obrázků, u kterých má odhadovaná třída stále relativně nízkou pravděpodobnost.

Krok 3 - společné trénování obou částí modelu

Doposud byli FeaModel a shlukovací hlava trénovány odděleně. Jelikož ale FeaModel může přiřazovat sémanticky podobné obrázky do různých shluků a shlukovací hlava má tendenci přiřazovat nepodobné vzorky do stejných shluků, je spoléhat na každou část modelu zvlášť sub-optimální. Autoři článku proto navrhli algoritmus pro důvěryhodnou anotaci, ve kterém se pro každý vzorek vybere určitý počet nejbližších sousedů z embeddingového prostoru získaného na konci předchozího kroku. Dále je zvolen prahový počet sousedů náležící ke stejné třídě, po jehož překonání bude tato třída přiřazena vzorku jakožto důvěryhodná.

Výsledkem algoritmu je tedy dataset rozdělený na dvě části. Jedna je podmnožina vstupních vzorků označená důvěryhodnou náležitostí ke třídě a druhá se sestává ze vzorků u nichž žádná ze tříd nedosáhla dostatečné důvěryhodnosti pro přiřazení. Tím se podařilo přeměnit problém shlukování bez dozoru na shlukování s částečným dozorem a obě části modelu mohou být dotrénovány metodou FixMatch [38].

3.2 SCAN - Semantic Clustering by Adopting Nearest neighbors

SCAN [16] je bývalý SOTA model, který byl překonán [1] novějšími modely jako například SPICE [31]. Do této práce byl vybrán z důvodu jeho nízké náročnosti na výpočetní hardware v porovnání s jeho nástupci (může být trénován i na jedné GPU).

Nevýhodou sítě je nutnost znalosti počtu shluků, které se ve vstupním datasetu mají nacházet. Autoři však ukázali, že pokud odhadneme tento počet i jako dvojnásobně větší, než je počet reálných tříd v datasetu, zůstává klasifikační přesnost prakticky stejná (zhorší se pouze v desetinách procenta či nízkých procentech), jako při zadání přesného počtu tříd.

3.2.1 Fáze trénování

Trénování modelu je rozděleno na tři částí. Nejdříve se síť učí smysluplně reprezentovat vstupní obrázky v embeddingovém prostoru. Výstupem tohoto kroku je pro každý vzorek seznam jeho n nejbližších sousedů. Následně dojde k sémantickému shlukování přičemž ztrátová funkce odměňuje přiřazení vzorku a jeho sousedů do jednoho shluku. Na závěr proběhne ladění vah modelu podle obrázků dosahující prahové jistoty přiřazení ke správnému shluku, kterou autoři stanovili na hodnotu 0.99.

Fáze Pretext - reprezentativní učení

V první části učení získá síť na vstupu obrázek z datasetu a vytvoří si jeho upravenou verzi (SCAN k úpravám používá například ořez, zrcadlení či změny jasu a kontrastu). Síť převede oba obrázky na body v embeddingovém prostory, přičemž ztrátová funkce odměňuje co nejmenší vzdálenost mezi těmito body. V průběhu učení je tedy minimalizací ztrátové funkce docíleno minimalizace vzdálenosti mezi embeddingy vstupního a upraveného obrázku. To pomáhá síti shlukovat obrázky podle významu abstraktních objektů na nich vyobrazených a ne podle nízkoúrovňových podobností.

Fáze Scan - sémantické shlukování

V této části učení dojde k získání n nejbližších sousedů pro každý vzorek (sousedé jsou získáni na základě vzdáleností v embedingovém prostoru vytvořeném v předchozí části učení) a učení probíhá za použití ztrátové funkce, která odměňuje klasifikování této skupiny sousedů do stejného shluku. Jinými slovy je odměňována maximalizace skalárního součinu vektorů náležitosti k jednotlivým třídám skupiny sousedů. Výstupní vrstva sítě v této fázi učení končí softmax funkcí a výstupní pravděpodobnosti udávají pravděpodobnosti příslušnosti vzorku k jednotlivým shlukům - skalární součin je tedy největší, pokud je všech n vzorků přiřazeno ke stejnému shluku a to s co nejvyšší pravděpodobností. Přestože autoři ukazují, že většina sousedních vzorků bývá součástí stejné sémantické třídy, nevyhnutelně se objeví případy, kdy tomu tak nebude a tyto falešně pozitivní příklady potom zhorší výsledky sítě. Proto bylo na závěr celého učení přidáno dotrénování pseudo-anotováním vzorků.

Fáze Selflabel - dotrénování pseudo-anotováním

Autoři experimentálně ověřili hypotézu, že vzorky s velmi vysokou pravděpodobností náležitosti k dané třídě ($p_{max} \approx 1$) jsou většinou klasifikovány ke správnému shluku (tyto vzorky dokonce mohou sloužit jako prototypy dané třídy). Proto se v této fázi učení pro výpočet ztrátové funkce používají už jen tyto vzorky s vysokou pravděpodobností, přičemž přeučení na tuto podmnožinu vzorků se předchází jejich silným upravením a použitím cross-entropy coby ztrátové funkce.

Vzorkům, jejichž pravděpodobnost náležitosti k dané třídě přesáhne stanovený limit až v průběhu této fáze trénování je také přiřazena pseudo-třída a v příští epoše jsou použity k výpočtu ztrátové funkce. Trénování pokračuje po zadaný počet epoch, přičemž se ale ukládají váhy pouze pokud model překoná počet pseudo-anotovaných vzorků minulého nejlepšího modelu.

3.3 RUC - Robust learning for Unsupervised Clustering

Přístup RUC [33] ke shlukování je založený na převzetí výsledku jiného modelu pro shlukování bez učitele coby zašumělého datasetu pro učení s částečným dohledem (semi-supervised learning). V tomto datasetu se RUC pomocí tří různých přístupů snaží najít spolehlivě označené vzorky, které poté použije ve vlastní trénovací fázi pro vylepšení výsledků předchozího modelu.

3.3.1 Nalezení spolehlivě označených vzorků

První fází běhu modelu je výše zmíněné hledání spolehlivě označených vzorků. Autoři modelu RUC porovnávají přístup založený na pravděpodobnostech náležitostí vzorků do svých tříd (získaných z výstupu předchozího shlukovacího modelu.), přístup založený na určení míry spolehlivosti označení podle označení nejbližších sousedů daného vzorku a kombinovaný přístup používající obě předešlé techniky.

Strategie založená na pravděpodobnostech

V tomto přístupu autoři použijí minimální práh pravděpodobnosti přiřazení k dané třídě pro výběr spolehlivě přiřazených vzorků. Motivace tohoto přístupu je daná tendencí shlukovacích modelů přiřazovat přílišnou jistotu vzorkům ležícím v embeddingovém prostoru

na pomezí více tříd. Proto je stanoven vysoký pravděpodobnostní práh, který vyfiltruje jen nejjistěji přiřazené vzorky pro následnou trénovací fázi. Nevýhodou tohoto přístupu je, že je kompletně závislý na do jisté míry nepřesném výstupu předchozího modelu.

Strategie založená na metrice

Tato strategie využívá nezávislou metriku získanou vlastním učením bez učitele (SimCLR [6]) k umístění vzorků do embeddingového prostoru. Každému vzorku je poté přiřazen daný počet nejbližších sousedů a výsledná třída, která je mezi jeho sousedy nejčastěji zastoupena (jako třídy sousedů se použijí pseudo-třídy přiřazené sousedům předchozím modelem). Vzorek je pak označen za důvěryhodný pouze pokud se takto získaná třída shoduje se třídou přiřazenou vzorku předchozím modelem.

Hybridní strategie

Při použití této strategie je vzorek brán jako důvěryhodně přiřazený pouze pokud jeho důvěryhodnost potvrdí obě výše popsané strategie.

3.3.2 Trénování s částečným dozorem pomocí spolehlivě označených vzorků

RUC rozdělí původní dataset na základě výstupu předchozí fáze algoritmu do dvou vzájemně disjunktních množin $\mathcal{U}$ a χ, kde $\mathcal{U}$ značí množinu nespolehlivě označených vzorků a χ značí množinu spolehlivě označených vzorků. Tyto množiny poté použije k vlastnímu trénování klasifikátoru. Při trénování používá paralelně dvě techniky - vanilla přístup a sofistikovaný přístup, které si mezi sebou vyměňují výsledné úpravy označení prvků, aby se zabránilo akumulaci chyb vzniklých kvůli nedostatkům každého přístupu zvlášť.

Vanilla přístup

Naivní řešení spočívá v použití algoritmu pro učení s částečným dozorem MixMatch [3], který počítá ztrátovou funkci zvlášť pro vzorky z každé množiny (cross-entropy [18] pro vzorky z χ a regularizace konzistence [11] pro $\mathcal{U}$). Tento algoritmus je použit přímo na množiny $\mathcal{U}$ a χ, jakoby množina χ opravdu představovala ground-truth realitu, kterou se snažíme modelovat.

Ve skutečnosti je však nutno počítat s určitou mírou nepřesně označených vzorků v χ, jejichž dopad se autoři snaží minimalizovat v následujících přístupech.

Sofistikovaný přístup

Prvním krokem sofistikovaného přístupu je vyhlazování označení vzorků, které spočívá v zavedení uniformního šumu do vektoru pravděpodobností náležitosti ke třídám každého vzorku z χ. Následně je hodnota ztrátové funkce vypočítána jako součet hodnot ztrátových funkcí cross-entropy nad prvky χ, cross-entropy nad silně upravenými vzorky χ a regularizace konzistence nad vzorky $\mathcal{U}$.

Paralelní trénování

Následně dojde k paralelnímu trénování dvou modelů RUC, z nichž jeden vyhodnocuje ztrátovou funkci vanilla přístupem a druhý používá sofistikovaný přístup. Oba modely v průběhu trénování započítávají do svých predikcí označení vzorků predikce druhého modelu metodou Co-refinement [27].

Na konci každé epochy pak dochází k aktualizaci množiny χ, do které jsou přiřazeny všechny prvky z $\mathcal{U}$, které aspoň u jednoho ze dvou trénovaných modelů přesáhly prahovou hranici jistoty.

3.4 Výběr modelu pro implementaci

Tabulka 3.1 ukazuje publikované výsledky modelů na datasetu CIFAR-10 [1], kde Přesnost udává procento obrázků zařazených do správného shluku a ARI a NMI jsou nejpoužívanější metriky pro měření míry informací obsažené ve shlucích (detailnější popis naleznete v kapitole 5).

Model	Přesnost	ARI	NMI
SPICE	91.8 %	0.836	0.850
RUC	90.3 %	-	-
SCAN	88.3 %	0.772	0.797

Tabulka 3.1: Porovnání úspěšností modelů na datasetu CIFAR-10

Nejlepších výsledků jasně dosahuje model SPICE. Jeho nevýhodou je však potřeba paralelního trénování na čtyřech GPU najednou a tudíž nutnost investice v řádech vysokých desítek až stovek tisíc korun na zprovoznění (ať už pro sestavení vlastního přístroje či využití cloudových výpočetních služeb). Na druhém místě je dle úspěšnosti model RUC (autoři nepublikovali výsledky pro metriky ARI a NMI), který dosáhl uvedené úspěšnosti vylepšením výsledku prvotního shlukování modelem SCAN.

Model SCAN z této trojice dosahuje nejslabších výsledků, ale jelikož jeho implementace vyžaduje pouze jednu GPU (což znamená, že by bylo možné ho zprovoznit a používat na dostupných výpočetních strojích v TFS bez nutnosti budování nového počítače s více GPU jednotkami či pronajímání výpočetního času ve výpočetních centrech) a jeho výstup je nutnou prerekvizitou pro trénování modelu RUC, rozhodli jsme se se zadavatelem projektu z TFS vybrat pro otestování použitelnosti shlukování bez učitele právě tento model. Pokud se prokáže jeho použitelnost na obrázcích z elektronových mikroskopů, bude dalším krokem použití nadstavby RUC a jednalo by se o pádný argument pro postavení výpočetního stroje schopného trénovat model SPICE.

Kapitola 4

Metody pro shlukování na základě metadat

V této kapitole shrnu vlastnosti tradičních metod pro shlukování bez učitele s tím, že se zaměřím spíše na klíčové myšlenky metod než na přesné detaily jejich definic. Z těchto metod na závěr vyberu algoritmy, jejichž úspěšnost budu porovnávat s úspěšností shlukování na základě obrazových dat. Jelikož o metadatech uložených s obrázky nemáme záměrně prakticky žádné informace (viz 6.2), obsahuje výčet širokou škálu algoritmů lišících se v přístupu ke shlukování.

4.1 K-means

Jedná se o algoritmus pro shlukování na základě rozdělování. Princip metody K-means [2] leží v rozdělování vzorků do k shluků na základě vzdáleností vzorků od těžiště každého shluku. Následující zjednodušený popis běhu algoritmu byl převzat z [2].

1. Náhodně vyber k středů shluků v prostoru.
2. Pro každý datový bod vypočítej vzdálenosti od středů shluků.
3. Přiřaď každý datový bod k jemu nejbližšímu středu shluku.
4. Vypočítej nové středy shluků na základě souřadnic všech datových bodů přiřazených ke stejnému středu v předchozím kroku.
5. Vypočítej vzdálenost datových bodů od nových středů a přiřaď je k těm nejbližším.
6. Ukonči algoritmus, pokud nedošlo ke změně přiřazení žádného datového bodu, jinak pokračuj od kroku 4.

4.2 K-medoids

Algoritmus K-medoids [23] je varianta algoritmu K-means, která se zaměřuje na nalezení nejvhodnějších medoidů pro každý shluk. Na rozdíl od K-means, kde jsou centroidy shluků mohou ležet kdekoliv v prostoru, algoritmus K-medoids označuje za střed shluku (tedy medoid) datový bod z datasetu, který leží nejblíže reálnému těžišti shluku. Algoritmus funguje následovně:

1. Náhodně vybereme k bodů z datasetu, které budou sloužit jako počáteční medoidy shluků.
2. Přiřaď každý bod k nejbližšímu medoidu a vypočítej cost-funkci (sumu vzdáleností mezi body a medoidem pro každý shluk).
3. Pro každý medoid m a každý další bod p v datasetu vypočítej změnu cost-funkce, která by nastala záměnou m a p (tzn. p by se stal medoidem namísto m).
4. Proveď nejvýhodnější záměnu, pokud snižuje celkovou sumu vzdáleností od medoidů v datasetu oproti její stávající hodnotě.
5. Pokud došlo k záměně, pokračuj od kroku 3, jinak ukonči algoritmus.

4.3 GenieClust

GenieClust [15] je hierarchický shlukovací algoritmus, jehož základní myšlenkou je spojování nejbližších shluků za účelem formování shluků nových. Prvním krokem je označení každého datového bodu jako shluku o jednom bodu. Následně v každé další iteraci dochází ke spojení nejbližších dvou shluků do shluku nového.

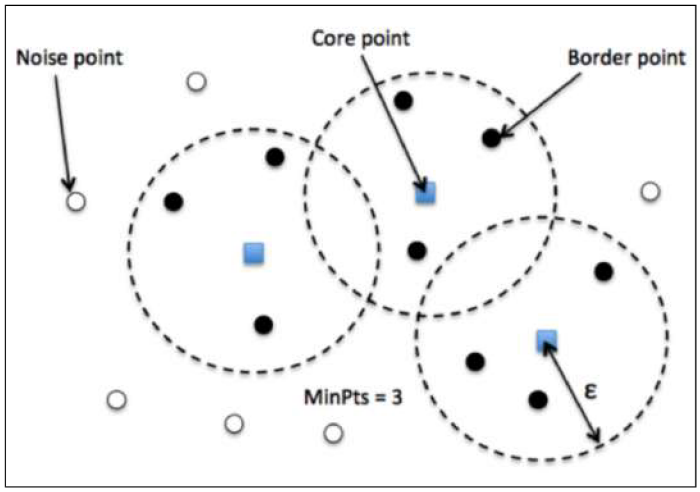
Aby se zamezilo tvorbě shluků diametrálně odlišných velikostí a následné degradaci do jednoho velkého shluku, ve chvíli kdy rozdíl ve velikostech shluků (měřený pomocí Gini indexu) přesáhne určenou hranici, dojde ke sloučení nejmenšího shluku s jeho nejbližším sousedem.

Kromě toho algoritmus GenieClust využívá také konsensuální shlukování, které se provádí opakovaným spuštěním algoritmu na různých podmnožinách dat a následným kombinováním výsledků. Tento přístup pomáhá snižovat vliv šumu na výsledné shluky.

4.4 DBSCAN

DBSCAN [12] je algoritmus pro shlukování na základě hustoty. Klíčovým konceptem je definování jádra shluku jakožto objektu, jehož ϵ-okolí obsahuje alespoň předem určený minimální počet dalších objektů, přičemž ϵ-okolím objektu se myslí okolí v n-rozměrném prostoru o poloměru ϵ. Objekt p je potom přímo dosažitelný z objektu q, pokud se p nachází v ϵ-okolí objektu q a q je jádro.

Tvorba shluků spočívá v nalezení jader databáze a přiřazení všech objektů přímo dosažitelných z každého jádra do jednoho shluku. Vzniká tak shluk jader a k nim přiřazených hraničních bodů, které už nejsou jádry a proto do shluku nemohou přiřadit body ze svého okolí, jak znázorňuje obrázek 4.1.



Obrázek 4.1: Znázornění shluku vytvořeného metodou DBSCAN. Převzato z [5].

Slabinou této metody je nutnost správného určení parametrů ϵ-okolí a minimálního počtu přímo dosažitelných objektů pro definování bodu jakožto jádra.

4.5 Affinity Propagation

Affinity propagation (AP) [13] se vyznačuje tím, že nevyžaduje předem definovaný počet shluků ani jiné předpoklady o struktuře dat. Na rozdíl od jiných algoritmů, které se snaží minimalizovat vzdálenosti mezi body, AP maximalizuje tzv. "přitažlivost"(afinitu) mezi datovými body, což umožňuje vytvářet shluky s různými tvarovými charakteristikami.

AP vytváří shluky posíláním zpráv mezi dvojicemi vzorků až do dosažení konvergence. Soubor dat je pak popsán pomocí malého počtu exemplářů, které jsou identifikovány jako ty, které nejvíce reprezentují ostatní vzorky.

Zprávy zasílané mezi dvojicemi vzorků udávají vhodnost jednoho být exemplářem druhého. Tato hodnota je aktualizována v reakci na zprávy z ostatních dvojic. Aktualizace probíhá iterativně až do konvergence, kdy jsou vybrány konečné exempláře, čímž je dáno konečné shlukování.

Mezi datovými body se vyměňují dva druhy zpráv a každá z nich zohledňuje jiný druh konkurence. Zodpovědnost $r(i,k)$ zasílaná z datového bodu i do kandidátního exemplárního bodu k, odráží nahromaděné důkazy o tom, jak vhodný je bod k jako exemplář pro bod i. Dostupnost $a(i,k)$ odeslaná z kandidátského exemplárního bodu k do bodu i odráží nahro-

maděné důkazy o tom, jak vhodné by bylo, aby si bod i vybral bod k jako svůj exemplář. Před výběrem exempláře dojde k vyhodnocení, jak vhodný jako exemplář ohodnocují bod k ostatní body a jak vhodné jsou pro bod i ostatní kandidáti na exemplární bod.

V průběhu algoritmu se iterativně aktualizují zodpovědnosti a dostupnosti pro každý bod, dokud algoritmus nedosáhne konvergence.

Nevýhodou algoritmu je nutnost nastavení parametru, který váhuje efekt nově obdržené zprávy na stávající stav náležitosti vzorku k exempláři.

4.6 Spectral Clustering

Spectral Clustering [30] vytváří nízko-rozměrné matice vzdáleností mezi vzorky a následně provádí shlukování (např. pomocí K-means) vlastních vektorů v nízko-rozměrném prostoru.

Současná verze Spectral Clusteringu vyžaduje, aby byl předem zadán počet shluků. Pro dva shluky řeší Spectral Clustering konvexní relaxaci problému normalizovaných řezů na grafu vzdáleností: rozřízne graf na dvě části tak, aby váha řezaných hran byla malá ve srovnání s vahami hran uvnitř každého shluku. Pro více shluků je tato metoda aplikována rekurzivně na vzniklé shluky. Práce algoritmu lze v principu rozdělit do následujících tří kroků:

1. Nejprve je z datového souboru vytvořena matice vzdáleností o rozměrech $n \times n$, kde n značí počet vzorků.
2. Z této matice je vypočtena Laplaceova matice a vlastní vektory.
3. Na závěr jsou data transformována do nového prostoru s menším počtem dimenzí pomocí vlastních vektorů Laplaceovy matice. Vlastní vektory jsou seřazeny podle velikosti příslušných vlastních hodnot a použity jako nové příznaky pro data. Následně je na tyto nová data aplikován standardní algoritmus shlukování jako například K-means.

Jednou z největších výhod této metody je schopnost zvládat data s neklasickými distribucemi, jako jsou např. zakřivené či propojené shluky, což je pro tradiční algoritmy shlukování často výzvou. Nevýhodou zůstává nutnost určení počtu shluků před samotnou analýzou.

4.7 Mean Shift

Cílem shlukování MeanShift [14] je objevit shluky ve skupině datových bodů s rovnoměrnou hustotou. Jedná se o algoritmus založený na centroidech, který aktualizuje kandidáty na centroidy tak, aby představovali těžiště bodů v dané oblasti. Tito kandidáti jsou následně filtrováni, aby se odstranily téměř duplicitní body a vytvořila se konečná sada centroidů.

Poloha kandidátů na centroidy se iterativně upravuje pomocí techniky zvané hill-climbing, která vyhledává lokální maxima odhadované hustoty pravděpodobnosti.

Výhodou Mean Shift je, že algoritmus sám odhadne počet shluků. Také dokáže odhalit shluky s různými tvary a velikostmi a je relativně robustní vůči šumu. Nicméně, Mean Shift trpí vysokou výpočetní složitostí, protože vyžaduje několikanásobné prohledávání nejbližších sousedů během provádění algoritmu a je náchylný k uvíznutí v lokálním maximu hustoty pravděpodobnosti, což může vést k vytváření falešných shluků.

4.8 Chameleon

Chameleon [22] pracuje s grafem, v němž uzly představují datové body a vážené hrany představují podobnosti či vzdálenosti mezi nimi. Algoritmus hledání shluků má následující hlavní fáze:

1. Nejdříve dojde k rozdělení grafu do velkého počtu relativně malých podshluků provedením řezů hran s nejmenší vahou (a tedy největší vzdáleností mezi uzly).

2. Následně je použit algoritmus aglomerativního hierarchického shlukování [39] k nalezení skutečných shluků opakovaným slučováním menších podshluků.

Klíčovou vlastností algoritmu je, že určuje dvojici nejpodobnějších podshluků s ohledem na vzájemnou propojenost i blízkost shluků. K tomu používá nový způsob měření míry vzájemné propojenosti a blízkosti mezi každou dvojicí podshluků, který bere v úvahu vnitřní charakteristiky samotných shluků. Není tedy závislý na nastavování parametrů uživatelem a dokáže se automaticky přizpůsobit vnitřním charakteristikám slučovaných dat.

Chameleon nepotřebuje před analýzou znát počet výsledných shluků, ale jeho výsledky jsou závislé na nastavení prahů blízkosti a propojenosti shluků pro sloučení.

4.9 BIRCH

BIRCH [42] je shlukovací metoda, která je koncipovaná pro použití na velkých datasetech. Bere v úvahu limitovanou paměť, která může být menší než paměť potřebná pro uložení všech původních dat.

BIRCH na začátku pro dataset sestaví Clustering Feature Tree (CFT), čímž dojde ke ztrátové kompresi dat do sady listových a nelistových CF uzlů. Každý CF uzel má své potomky a reprezentuje podshluk. Potomky listových uzlů ale mohou být jen reálné datové body reprezentované embeddingovými vektory a už ne další CF uzly. Pokud CF uzel není listový, tak jeho potomci reprezentují další podshluky. Ke každému listu jsou uloženy následující agregované informace:

- Počet vzorků v podshluku.
- Lineární součet embeddingových vektorů vzorků v podshluku.
- Součet druhých mocnin embeddingových vektorů vzorků v podshluku.

Algoritmus tedy umožňuje redukovat vstupní data na množinu podshluků, které jsou získány z listů CFT. Pokud je zadán parametr požadovaného počtu výsledných shluků (a pokud se tento počet liší od počtu listů CFT), je na závěr celý strom zpracovaný globálním shlukovačem.

Algoritmus BIRCH má dva klíčové parametry - faktor větvení a práh. Faktor větvení určuje maximální počet potomků (podshluků) v uzlu, zatímco práh určuje maximální vzdálenost, kterou může mít nově vkládaný vzorek od existujícího vzorku v listovém uzlu, aby došlo k absorpci nového vzorku.

Popis algoritmu:

1. Nový vzorek je vložen do kořenového uzlu CFT, ze kterého putuje stromem vždy do nejbližšího potomka daného uzlu. Jakmile dosáhne listového uzlu, najde nejbližší datový bod a dojde k testu, zda splňuje prahovou (vzdálenostní) podmínku. Pokud ano, je nový vzorek absorbován (dojde pouze k aktualizaci agregovaných informací uložených v uzlu), pokud ne, je vektor vzorku uložen jako další potomek listového uzlu.

2. Dojde ke kontrole počtu potomků. Pokud po přidání nového vzorku obsahuje listový uzel více potomků, než dovoluje faktor větvení, dojde k vytvoření nového nadřazeného uzlu a rozdělení původního listu na dva nové listy.

3. Následně se rekurzivně aktualizují agregované informace nadřazených uzlů. Pokud došlo ke vzniku nového uzlu, může v tomto kroku dojít k dalším dělení nadřazených uzlů.

4. Pokud došlo k rozdělení uzlu, dojde na závěr ke kontrole prvního již nerozděleného uzlu na cestě od nového vzorku ke kořeni. Pokud tento uzel obsahuje dva potomky, které mohou být sloučeny aniž by byl ve výsledném uzlu překročen faktor větvení, dojde ke sloučení.

Jakmile byly do stromu vloženy všechny prvky, může dojít ke globálnímu shlukování do zadaného počtu shluků pomocí aglomerativního shlukování [34] na základě agregovaných informací uložených v uzlech.

4.10 Výběr algoritmů pro implementaci

Jak je uvedeno na začátku kapitoly, o datech, nad kterými bude prováděno shlukování, nemáme záměrně prakticky žádné informace. Zároveň nejsou kladeny žádné požadavky na časovou složitost shlukovacího algoritmu a velkou váhu nemá ani nevýhoda některých algoritmů spočívající v nutnosti zadání počtu shluků, jelikož bude vždy k dispozici odborný odhad počtu tříd pro shlukování.

Naopak důležité pro výběr shlukovacího algoritmu bude jeho odolnost vůči odlehlým hodnotám a použitelnost na vysoce dimenzionálních datech, přičemž díky relativně nízkému počtu vzorků v řádech stovek nemusíme brát v potaz limitaci fyzickou pamětí výpočetního stroje.

Na základě výše uvedených argumentů jsem se rozhodl vybrat takové algoritmy, které nejsou závislé na ladění svých parametrů podle specifických vlastností datasetu, na který mají být použity (jako například DBSCAN) ani náchylné k ovlivnění odlehlými hodnotami (jako například K-Means). S přihlédnutím k informacím o úspěšnostech některých shlukovacích algoritmů [36, 15] a ve snaze vybrat algoritmy, které ke shlukování používají odlišné přístupy byly nakonec vybrány algoritmy K-medoids, Affinity Propagation, Spectral Clustering a GenieClust.

Kapitola 5

Metriky měření úspěšnosti shlukování

Hodnocení výkonnosti shlukovacího algoritmu je komplexnější než hodnocení klasifikačních algoritmů - triviální metriky jako je počet chyb po porovnání výstupu s ground-truth labely nelze použít, jelikož přiřazení názvů tříd k jednotlivým shlukům je náhodné. Při hodnocení shlukovacích algoritmů je tedy třeba nejdříve nalézt zobrazení mezi výstupními shluky a ground-truth labely, které minimalizuje počet chybných přiřazení, a až následně lze použít samotné metriky pro měření úspěšnosti. K nalezení výše zmíněného zobrazení je v této práci použit Maďarský algoritmus [26].

V této kapitole shrnu vlastnosti nejčastěji používaných metrik pro vyhodnocování úspěšnosti shlukovacích algoritmů a na závěr vyberu metriky, které použiji k vyhodnocení modelů testovaných v této práci.

5.1 Přesnost

Princip přesnosti algoritmu spočívá v jednoduchém výpočtu poměru vzorků přiřazených ke správné třídě (a tudíž umístěných do správného shluku) [29]. Přesnost *accuracy* tedy získáme následující rovnicí:

$$\text{accuracy}(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} 1\left(\hat{y}_i = y_i\right),$$

kde n udává počet vzorků, $\hat{y}$ je vektor labelů přiřazených vzorkům a y vektor ground-truth labelů. $1\left(\hat{y}_i = y_i\right)$ je pak rovno 1 pokud platí $\hat{y}_i = y_i$, jinak je rovno 0.

Byť je přesnost důležitou informací, sama o sobě může být zavádějící - neodhalí totiž náhodné označení vzorků, z nichž část se bude s určitou pravděpodobností shodovat s ground-truth labely a může tedy vracet dobrou úspěšnost algoritmu, jehož výstup je ve skutečnosti náhodný. Podobně v případě vyhodnocování shlukování silně nevyváženého datasetu může model, který přiřadí všechny vzorky do jediné třídy, dosahovat vysoké přesnosti.

5.2 Random Index a Adjusted Random Index

Adjusted Rand Index (ARI) [21] je založený na metrice Rand Index (RI), jejíž základní myšlenka je následující - pro množinu prvků $S = \{o_1, \ldots, o_n\}$ a její rozdělení $X = \{X_1, \ldots, X_r\}$, které rozděluje S do r shluků, a $Y = \{Y_1, \ldots, Y_s\}$, které rozděluje S do s shluků, lze definovat následující hodnoty:

- $a = |S^*|$, kde $S^* = \{(o_i, o_j) \mid o_i, o_j \in X_k, o_i, o_j \in Y_l\}$
- $b = |S^*|$, kde $S^* = \{(o_i, o_j) \mid o_i \in X_{k_1}, o_j \in X_{k_2}, o_i \in Y_{l_1}, o_j \in Y_{l_2}\}$
- $c = |S^*|$, kde $S^* = \{(o_i, o_j) \mid o_i, o_j \in X_k, o_i \in Y_{l_1}, o_j \in Y_{l_2}\}$
- $d = |S^*|$, kde $S^* = \{(o_i, o_j) \mid o_i \in X_{k_1}, o_j \in X_{k_2}, o_i, o_j \in Y_l\}$

pro $1 \leq i, j \leq n, i \neq j, 1 \leq k, k_1, k_2 \leq r, k_1 \neq k_2, 1 \leq l, l_1, l_2 \leq s, l_1 \neq l_2$.

RI pak lze vypočítat následující rovnicí, která udává procentuální poměr shod mezi X a Y ze všech možných párů prvků z množiny S:

$$RI = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}.$$

Výsledná hodnota RI je jedna pro identické rozdělení X a Y a nula pro taková rozdělení, která se neshodují v žádném páru.

ARI je pak nadstavba metriky Rand Index (RI) upravená tak, aby do výsledného ohodnocení byla započtena možnost shod náhodným shlukováním. Výpočet ARI je založen na kontingenční tabulce shod mezi X a Y, kde každé n_{ij} značí počet shodných vzorků ve shlucích X_i a $Y_j : n_{ij} = |X_i \cap Y_j|$.

X	Y_1	Y_2	$\cdots$	Y_s	sumy
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
sumy	b_1	b_2	$\cdots$	b_s	

Výsledné skóre ARI je potom vypočteno následující rovnicí:

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}.$$

Na rozdíl od RI tak ARI může nabýt záporné hodnoty pro dvojici rozdělení, která dosahuje horší, než náhodné úspěšnosti ve shodě shluků. Horní hranice skóre pro shodné množiny shluků však zůstává stejná jako u RI.

5.3 Normalized Mutual Information

Normalized Mutual Information (NMI) [40] bere dvě rozdělení U a V vytvořená z množiny prvků N a měří míru jejich vzájemně sdílených informací (tedy kolik informace o rozdělení V získáme pokud známe rozdělení U). Pojem sdílené informace je v tomto kontextu úzce spjatý s entropií (tedy očekávaným množstvím informace náhodného rozdělení). Entropií je v tomto kontextu množství nejistoty definované pro U následující rovnicí:

$$H(U) = -\sum_{i=1}^{|U|} P(i)\log(P(i)),$$

kde $P(i) = |U_i|/N$ je pravděpodobnost, že vzorek náhodně vybraný z U spadá do třídy U_i. Obdobná rovnice definuje výpočet entropie pro V:

$$H(V) = -\sum_{j=1}^{|V|} P'(j)\log\left(P'(j)\right),$$

kde $P'(j) = |V_j|/N$.
Mutual Information (MI) mezi U a V lze získat následujícím způsobem:

$$\mathrm{MI}(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P(i,j)\log\left(\frac{P(i,j)}{P(i)P'(j)}\right),$$

kde $P(i,j) = |U_i \cap V_j|/N$ je pravděpodobnost, že náhodně vybraný vzorek spadá jak do třídy U_i, tak do třídy V_j.

Metriku NMI získáme normalizací výsledné MI do intervalu $< 0, 1 >$ následovně:

$$\mathrm{NMI}(U,V) = \frac{\mathrm{MI}(U,V)}{\mathrm{mean}(H(U),H(V))}.$$

5.4 Výběr metrik pro měření úspěšnosti shlukování

Všechny v této kapitole popsané metriky se v literatuře standardně používají pro porovnávání úspěšnosti shlukovacích algoritmů [25]. Některé zdroje uvádí, že ARI je metrika vhodnější pro případ, kde ground-truth jsou shluky vyvážených velikostí a NMI je vhodnější pro ground-truth obsahující shluky různých velikostí [37]. Jelikož firma TFS dodala k otestování jak vyvážené, tak nevyvážené datasety a jelikož Přesnost je intuitivní metrika používaná při vyhodnocování všech algoritmů v literatuře, rozhodl jsem se ve své práci použít Přesnost, ARI i NMI pro získání co nejkompletnější představy o kvalitě výsledných shluků jednotlivých algoritmů.

Kapitola 6

Datasety obrázků z elektronové mikroskopie

Trénování a testování modelů pro shlukování jsem provedl na několika datasetech obrázků z elektronových mikroskopů firmy TFS.

V této kapitole stručně popíši obecné vlastnosti těchto obrázků a jejich metadat, proces výběru obrázků do datasetu a jejich anotace za účelem testování úspěšnosti shlukovacích modelů. Na závěr pak rozeberu rozdíly mezi datasety z TFS a standardními datasety pro porovnávání úspěšnosti modelů.

6.1 Obrazová data

Obrázky z elektronových mikroskopů se liší pro různé fáze běhu mikroskopu a různé typy materiálu, se kterým mikroskopy pracují. Všechny mají ale několik obecných vlastností společných - jsou ukládány ve stupních šedi a často ve velmi vysokém rozlišení (řádově až 2000 x 2000 pixelů). Fotografovaný objekt (často lamela v různých fázích práce mikroskopu nebo povrch materiálu v různých fázích vyřezávání lamely) je umisťován zhruba do středu obrázku a často zabírá jen malou část prostoru fotografie.

Tyto obecné informace budou užitečné při vybírání transformací pro první fázi učení (viz 2.1) vybraného modelu pro shlukování podle obrazových dat.

6.2 Metadata

Ke každému obrázku jsou ukládány údaje o nastavení mikroskopu v době jeho pořízení. Jedná se 100 až 200 intervalových a kategorických proměnných, které popisují jak nastavení použitá při pořízení obrázku, tak hodnoty senzorů monitorujících vnitřní stav mikroskopu a údaje o fázi běhu mikroskopu, ve které byl obrázek pořízen. Specifický soubor údajů, který se k obrázkům ukládá, se mění nejen v závislosti na specifickém mikroskopu, na kterém byl obrázek pořízen a v závislosti na fázi běhu, ale také v závislosti na verzi softwaru řídícího běh mikroskopu.

Jelikož firma TFS požaduje, aby bylo možné algoritmus použít na jakoukoliv podmnožinu obrázků pořízených na všech mikroskopech za posledních několik let, bylo by nutné analyzovat pravděpodobně vyšší stovky typů údajů, na která lze v metadatech narazit (a i tak by to bylo nedostatečné řešení, jelikož by nepočítalo s novými metadaty, které se k obrázkům mohou přidat v budoucnu).

Na základě těchto skutečností, jsem se rozhodl neanalyzovat metadata ručně za účelem snížení dimensionality dat, ale při každém běhu algoritmu vytvořit automaticky databázi všech typů metadat, které jsou uložené aspoň k jednomu obrázků ze shlukovaného datasetu. Argumentem proti vyfiltrování irelevantních údajů z metadat před samotným shlukováním je také to, že účelem shlukování je možnost nalezení korelace mezi specifickou kombinací nastavení datasetu a chybou při běhu mikroskopu. Zmiňované filtrování by však bylo založeno na předešlých znalostech odborníků z TFS a mohlo by zabránit nalezení nových a nečekaných informací.

6.3 Anotace datasetů

Obrázky z datasetů byly s asistencí odborníků z TFS rozděleny do tříd na základě vizuální podobnosti takovým způsobem, aby rozdělení odpovídalo požadovanému výsledku po použití programu na datasetu v praxi.

Pro testování různých přístupů ke shlukování bylo vybráno několik datasetů, které lze je rozdělit do dvou skupin:

1. Datasety obsahující obrázky ze stejného kroku práce mikroskopu. Obrázky jsou v ideálním případě prakticky identické s prakticky identickými metadaty, jediný rozdíl by měly způsobovat chyby v procesu na obrázku zachycené.

2. Datasety obsahující směs obrázků z většího počtu různých kroků práce mikroskopu. Tyto obrázky vetšinou lze rozdělit na základě vizuální podobnosti na značně nižší počet tříd, než je původní počet různých kroků.

6.3.1 Needle Attachment Large a Needle Attachment Small

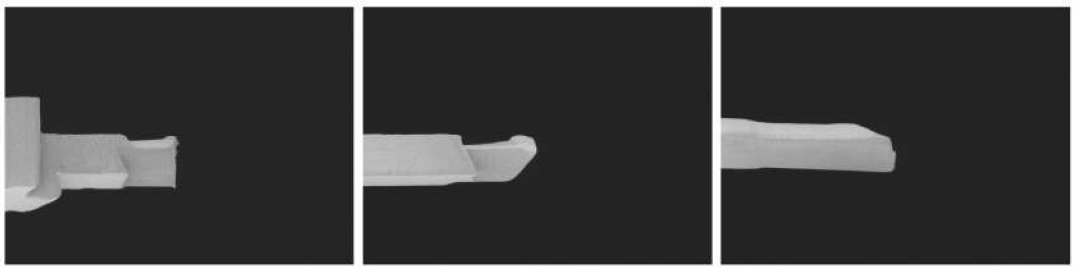
Dataset Needle Attachment Large (dále jen NAL) se sestává ze 3040 obrázků sesbíraných v rámci několika běhů mikroskopu na podobných vzorcích. Obrázky jsou tedy z více běhů, ale všechny byly pořízeny ve stejné fázi práce mikroskopu s názvem Needle Attachment. Celý proces se sestává z celkem 233 kroků, ve kterých mikroskop pořizuje obrázky. Všechny tyto obrázky lze ale rozdělit do následujících tří hlavních sémantických skupin s odpovídajícím zastoupením v datasetu:

- Kontrola jehly z úhlu (610 obrázků). Zástupci třídy jsou vyobrazeni na obrázku 6.1.
- Kontrola jehly v horizontální pozici (605 obrázků). Zástupci třídy jsou vyobrazeni na obrázku 6.2.
- Kontrola místa výřezu lamely (1825 obrázků). Zástupci třídy jsou vyobrazeni na obrázku 6.3.

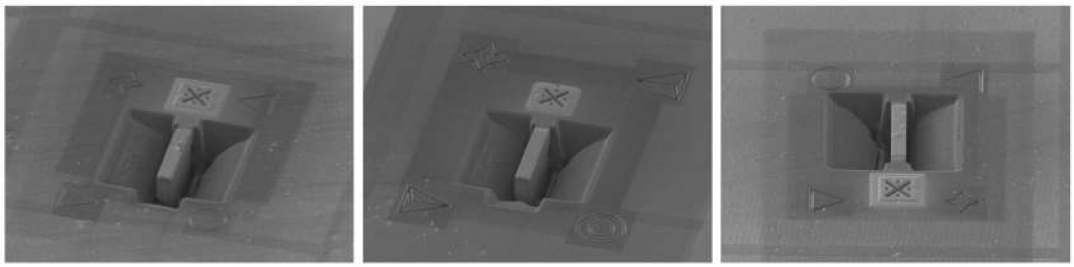
Dataset byl vybrán odborníky z TFS, jelikož kontrola práce několika automatizovaných běhů mikroskopu na podobných vzorcích je často prováděnou činností v rámci testování softwaru pro automatizaci mikroskopů. Dataset slouží jako reprezentant datasetů kategorie 2.

Dataset Needle Attachment Small (dále jako NAS) je pak podmnožinou obrázků z datasetu NAL. NAS se sestává ze stejných tří tříd jako NAL, ale každá třída obsahuje obrázky z pouze jednoho korku mikroskopu. Třídy jsou navíc perfektně vyvážené, jelikož každá z nich se sestává přesně ze 52 obrázků. NAS byl vytvořen ze dvou důvodů - za prvé pro porovnání s výsledky shlukování NAS a vyhodnocení vlivu nižšího počtu obrázků na učení a za druhé za účelem provedení ideálního experimentu pro shlukování podle metadat, jehož cílem je ověřit, jestli má tento typ shlukování naději dosáhnout použitelné úspěšnosti.

Obrázek 6.1: Příklad obrázků pro kontrolu jehly z úhlu v datasetech NAS a NAL.



Obrázek 6.2: Příklad obrázků pro kontrolu jehly v horizontální pozici v datasetech NAS a NAL.

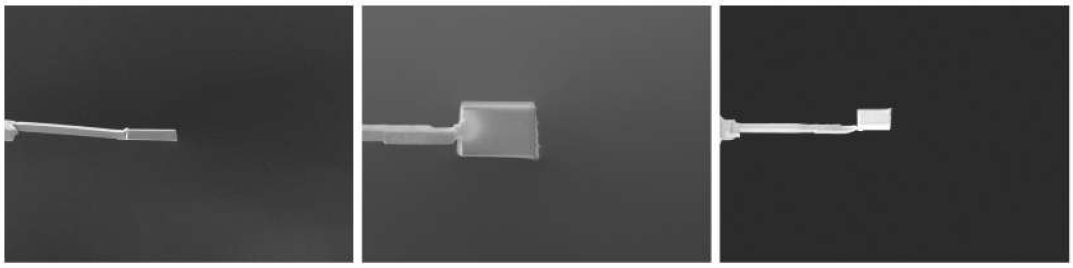


Obrázek 6.3: Příklad obrázků pro kontrolu místa výřezu lamely v datasetech NAS a NAL.

6.3.2 Welding Manipulator

Tento dataset obsahuje obrázky z jediného kroku práce mikroskopu, mezi kterými se vyskytují obrázky s chybou (spadá tak do kategorie 1). Dataset tedy tvoří dvě nevyvážené třídy - třída obrázků bez chyby čítající 40 obrázků a třída obrázků s chybou čítající 7 obrázků. V tomto kroku práce mikroskopu dochází k přesunu lamely z místa výkopu na místo pro navaření (tzv. prst) pomocí transportní jehly. Tento krok byl záměrně vybrán, jelikož obsahuje pro obrazové rozpoznávání jednoduchý typ chyby - v průběhu přenosu může lamela z jehly odpadnout, což se projeví tím, že na obrázku bude zachycena samotná jehla. Problém je tedy v odlišení obrázků s jehlou a výrazným obdélníkem (lamelou) uprostřed (viz obrázek 6.4) a těch pouze s jehlou (viz obrázek 6.5).

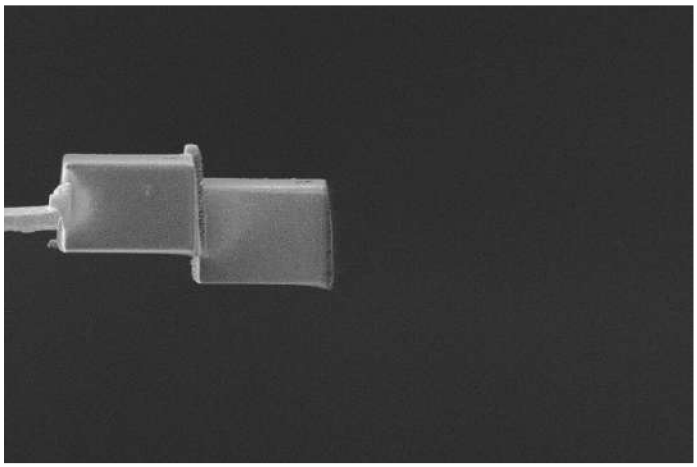
Reálně v tomto kroku nastává ještě druhý typ chyby, jelikož může na transportní jehle zůstat přivařená lamela z předchozího běhu, což má za následek přivaření nové lamely na konec lamely staré (viz 6.6). Za účelem zjednodušení už tak náročného úkolu pro shlukovací algoritmy jsme se ale s vedoucím z TFS rozhodli brát obrázky se zdvojenými lamelami jako korektní s tím, že pokud by algoritmy na tomto datasetu dosahovali dobré úspěšnosti, byla by vytvořena jeho náročnější verze s oběma druhy chyb.



Obrázek 6.4: Příklad obrázků z datasetu Welding Manipulator s lamelou na transportní jehle.

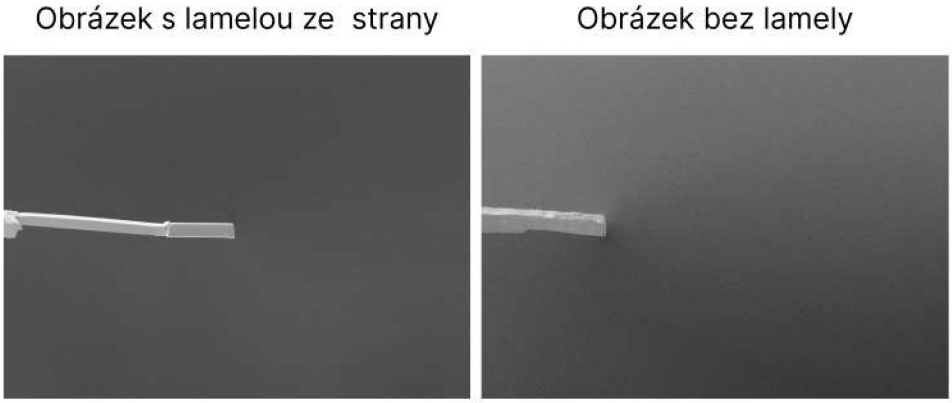
Obrázek 6.5: Příklad obrázků z datasetu Welding Manipulator bez lamely, která pravděpodobně upadla z transportní jehly.

Tento dataset byl vybrán, jelikož chyba, ke které může v tomto kroku dojít je relativně lehce rozpoznatelná oproti chybám vyskytujícím se v jiných krocích (viz 6.9). I tak se ale jedná o velmi náročné obrázky na rozlišení pomocí učení bez učitele. Největší důvod ke skepticismu vůči úspěšnosti výsledného shlukování jsou obrázky s jehlou a lamelou z profilu (viz 6.7), kde je na první pohled i pro člověka těžko rozpoznatelné, zda se nejedná pouze o samotnou jehlu.



Obrázek 6.6: Příklad obrázků z datasetu Welding Manipulator se zdvojenou lamelou na transportní jehle.

U jiných druhů chyb ale bývá informace o chybě ukryta i v mnohem méně výrazném detailu - třeba v mírném zakřivení či natočení lamely nebo v pozici jehly vůči lamele, jak je znázorněno na obrázku 6.9.



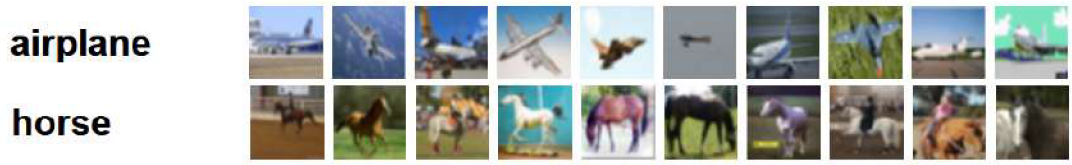
Obrázek 6.7: Příklad těžké rozpoznatelnosti přítomnosti lamely na jehle u obrázků z datasetu Welding Manipulator.

6.4 Porovnání datasetů z TFS se standardními datasety pro měření úspěšnosti modelů

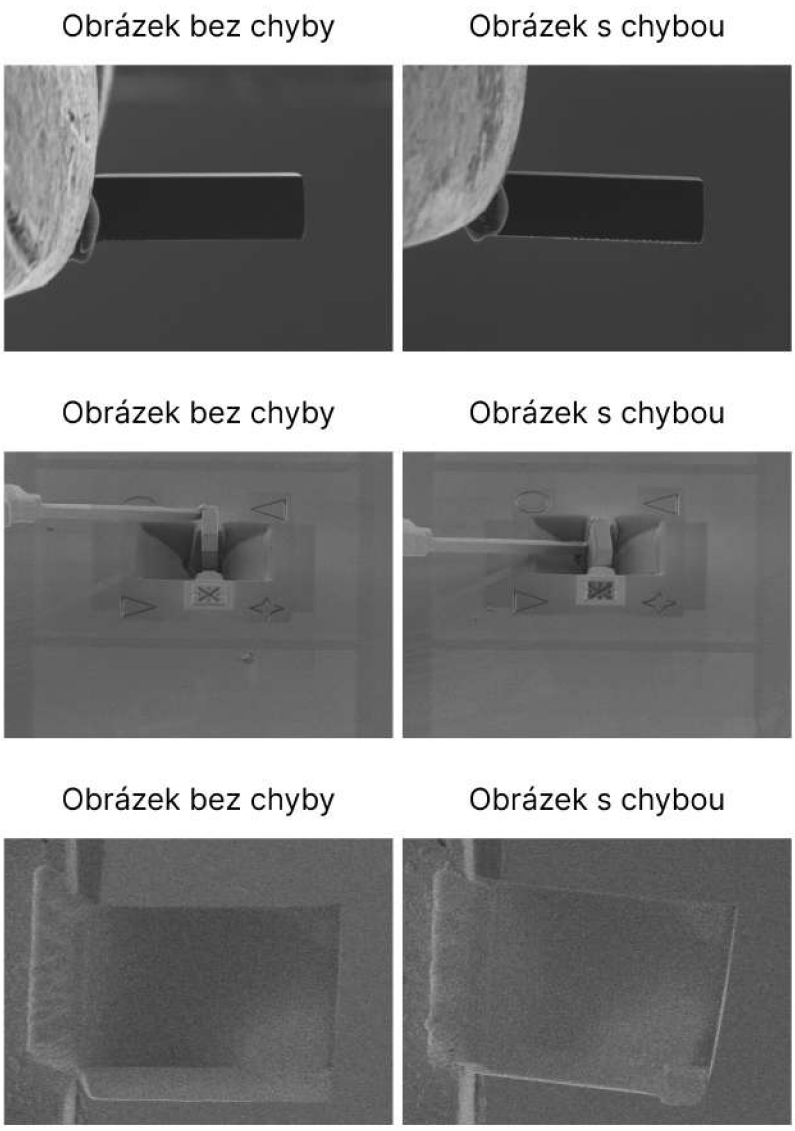
Modely pro shlukování obrázků se kterými pracuji byly testovány na datasetech CIFAR10, CIFAR20-100 a STL10, které jsou velmi odlišné od datasetů obrázků z TFS. Tyto datasety obsahují obrázky v rozlišení 32x32 pixelů se třemi barevnými kanály. Objekty určující třídu obrázku zabírají většinu jeho plochy a jsou umístěný zhruba do středu s tím, že pozadí obrázku mnohdy koreluje s daným objektem (například obrázky třídy "kůň"často obsahují koně na pozadí zelené louky, kdežto obrázky třídy "letadlo"často obsahují letadla na pozadí modré oblohy či asfaltové ranveje 6.8).

Obrázky z TFS jsou naopak ve vysokém rozlišení ale pouze v odstínech šedi (tedy s jedním barevným kanálem) a pozadí obrázků různých tříd jsou často velmi podobná (viz 6.9). Navíc objekt, který určuje třídu obrázku často zabírá jen malou část jeho plochy a je důvod k obavě, že ořezávání a vyřezávání děr v obrázcích, což jsou úpravy standardně používané při trénování modelů pro shlukování bez učitele, mohou celý objekt vystřihnout a tím vnášet do modelu nejistotu.

Právě tyto rozdíly jsou důvodem ke skepticismu vůči dosažení publikované úspěšnosti z datasetu CIFAR10 i na datasetech z TFS.



Obrázek 6.8: Příklad obrázků ze tříd letadlo a kůň datasetu Cifar-10. Převzato z [24].



Obrázek 6.9: Příklad podobnosti obrázků s chybou a bez chyby z TFS.

Kapitola 7

Popis výsledného systému pro shlukování obrázků a testování různých nastavení modelu SCAN

Za účelem naplnění cílů této práce jsem v jazyce Python implementoval systém, který slouží k experimentování na datasetech z TFS s vybranými modely pro shlukování. Systém má tři základní funkce:

1. Porovnávání výsledků různých shlukovacích metod pomocí metrik Přesnost, ARI a NMI na anotovaných datasetech z TFS.
2. Testování vlivu změn různých parametrů modelu SCAN na jeho úspěšnost (podle výše zmíněných metrik).
3. Použití v praxi - tedy ke shlukování neoznačených obrázků z TFS pomocí zvolené metody.

7.1 Implementace shlukovacích algoritmů

Ve své práci jsem použil volně dostupnou implementaci modelu SCAN, kterou autoři zveřejnili ve svém repositáři na GitHubu [41]. Tuto implementaci jsem do jisté míry upravil, aby fungovala na obrázcích z TFS a rozšířil o vlastní sadu transformací, jejíž efekt na úspěšnost modelu jsem se rozhodl otestovat.

Pro shlukování podle metadat jsem implementoval algoritmus, který extrahuje metadata z datasetu obrázků do podoby MySQL databáze a intervalové atributy databáze poté normalizuje do intervalu $< 0, 1 >$. Pro použití vybraných shlukovacích algoritmů na tuto databázi jsem vybral implementaci z knihovny SciKit [8].

7.2 Systém pro shlukování obrázků

Systém pro shlukování na vstupu přijme cestu k datasetu, nastavení parametrů pro model SCAN, seznam metod pro shlukování podle metadat, které má použít a informaci, zda má provést reálné shlukování neoznačených obrázků, nebo jen vyhodnotit úspěšnosti shlukovacích metod na anotovaném datasetu.

Oproti publikované verzi jsem v modelu SCAN přidal měření úspěšnosti po první pretextové fázi učení (aplikováním shlukovacího algoritmu K-medoids na výsledné embeddingové vektory). Na závěr každé fáze učení jsou vykresleny grafy popisující průběhy hodnot ztrátové funkce a hodnot metrik pro měření úspěšnosti v průběhu učení.

Po dokončení celého trénování jsou vykresleny grafy porovnávající výsledky jednotlivých fází učení modelu SCAN s metodami pro shlukování podle metadat. Tato měření se samozřejmě provádí jen v případě spuštění algoritmu na anotovaném datasetu za účelem vyhodnocení úspěšností shlukovacích metod.

Při spuštění programu pro reálné shlukování neoznačených obrázků dojde k vytvoření adresářů pro každou použitou shlukovací metodu. V každém adresáři je pak vytvořen počet podadresářů odpovídající počtu tříd, do kterých chce uživatel dataset rozdělit. Do každého z těchto podadresářů jsou pak nakopírovány obrázky z původního datasetu přiřazené do odpovídající třídy danou shlukovací metodou.

7.3 Systém pro testování různých nastavení modelu SCAN

Systém pro testování různých nastavení modelu SCAN spouští trénování modelu s různými hodnotami rozlišení vstupních obrázků, různými počty epoch, transformačních sad a různými přístupy k řešení problémů spojených s fází učení Selflabel. Celý proces včetně výběru těchto parametrů je popsán detailněji v kapitole 8.

Pro každou kombinaci parametrů modelu SCAN je po dokončení trénování uložena výsledné hodnoty metrik pro měření úspěšnosti.

Kapitola 8

Testování různých nastavení modelu SCAN

Při prvotních experimentech s modelem SCAN se mi na datasetech z TFS nepodařilo dosáhnout publikované úspěšnosti. Proto jsem se rozhodl ověřit několik hypotéz týkajících se příčin těchto špatných výsledků otestováním různých nastavení parametrů modelu. Testování jsem provedl na datasetu Needle Attachment Small 6.3.1, jelikož na tomto datasetu model SCAN dosahoval už v prvotních experimentech alespoň použitelné úspěšnosti, zatímco na datasetu Welding Manipulator 6.3.2 docházelo k prakticky náhodnému shlukování. počet obrázků v datasetu Needle Attachment Small také umožňuje v časovém horizontu několika dní provést trénování vyššího počtu různých nastaveních modelu, které by na datasetu Needle Attachment Large zabralo minimálně několik týdnů.

V této kapitole popíši tyto hypotézy týkající se příčin horších výsledků modelu SCAN na datasetech z TFS a shrnu závěry z experimentů provedených za účelem jejich vyhodnocení.

8.1 Výběr parametrů pro testování

Parametrů pro nastavení neuronové sítě je takové množství, že snažit se otestovat jen několik různých hodnot pro každý z nich (a tudíž všechny kombinace těchto hodnot) není z důvodu časové náročnosti možné. Proto jsem na základě znalostí rozdílů mezi datasety z TFS a datasetem CIFAR-10, na němž byly předvedeny dobré výsledky modelu SCAN [1], vybral následující parametry, které by mohly úspěšnost modelu na datasetech z TFS ovlivnit nejvíce.

8.1.1 Transformace modelu SCAN

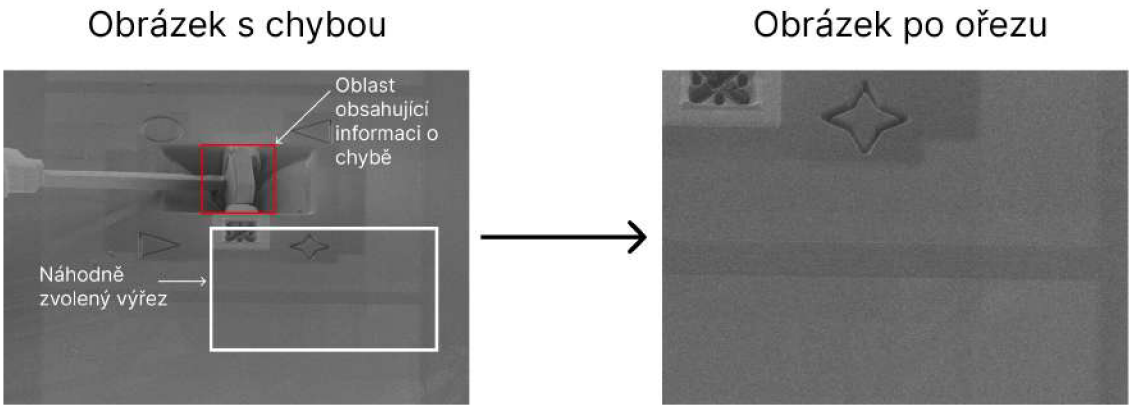
První hypotézou je, že transformace použité autory modelu mohou mít negativní vliv na učení na datasetech z TFS a to zejména na těch obsahujících obrázky z jednoho kroku práce mikroskopu s chybou a bez chyby, ve kterých často objekt určující třídu obrázku zabírá jen zlomek celkové plochy (jak bylo ukázáno v kapitole 6.4). Pokud není uvedeno jinak, jsou všechny transformace převzaty z knihovny torchvision.transforms [9].

Autoři použili pro trénování následující transformace s tím, že pro normalizaci počítají průměr a standardní odchylku ze všch hodnot pixelů všech obrázků z datasetu):

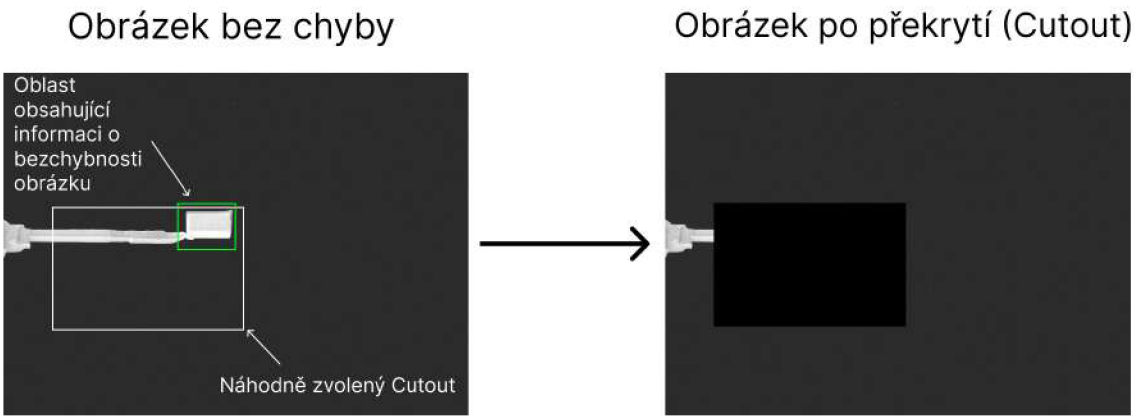
- Transformace obrázků pro pretextovou fázi učení modelu SCAN
  - `RandomResizedCrop()`: Náhodně ořízne obrázek a změní velikost výsledného ořezu na základě specifikovaných parametrů.
  - `RandomHorizontalFlip()`: Převrátí obrázek horizontálně, k čemuž dojde se zadanou pravděpodobností.
  - `ColorJitter()`: Náhodně upraví hodnoty jasu, kontrastu, sytosti a odstínu obrázků.
  - `RandomGrayscale()`: Převede obrázek do stupňů šedi, k čemuž dojde se zadanou pravděpodobností.
  - `Normalize()`: Provede normalizaci obrázků podle zadaného průměru a standardní odchylky.
- Transformace obrázků pro fáze Scan a Selflabel učení modelu SCAN
  - `RandomHorizontalFlip()`: Převrátí obrázek horizontálně, k čemuž dojde se zadanou pravděpodobností.
  - `RandomCrop()`: Na náhodném místě ořízne obrázek na zadanou velikost.
  - `Augment()`: Náhodně vybere a provede zadaný počet transformací obrázku z následujícího seznamu transformací implementovaných autory modelu:
    * `Identity()`: Tato transformace vrátí obrázek nezměněný.
    * `AutoContrast()`: Tato transformace upravuje kontrast obrázku tak, aby nejtmavší a nejsvětlejší pixely byly namapovány na hodnoty 0 a 255.
    * `Equalize()`: Tato transformace upravuje kontrast obrázku vyrovnáním histogramu intenzit pixelů.
    * `Rotate()`: Otočí obrázek o náhodný úhel mezi -30 a 30 stupni.
    * `Solarize()`: Invertuje všechny pixely, jejich hodnota přesahuje určenou prahovou hodnotu.
    * `Color()`: Náhodně upraví saturaci obrázku o náhodný faktor mezi 0,05 a 0,95.
    * `Contrast()`: Náhodně upravuje kontrast obrázku o náhodný faktor mezi 0,05 a 0,95.
    * `Brightness()`: Náhodně upraví jas obrázku o náhodný faktor mezi 0,05 a 0,95.
    * `Sharpnes()`: Náhodně upraví ostrost obrázku o náhodný faktor mezi 0,05 a 0,95.
    * `ShearX()`: Aplikuje horizontální zkosení obrázku o náhodný faktor mezi -0,1 a 0,1.
    * `TranslateX()`: Aplikuje horizontální posun obrázku o náhodný faktor mezi -0,1 a 0,1.
    * `TranslateY()`: Aplikuje vertikální posun obrázku o náhodný faktor mezi -0,1 a 0,1.

    - `Posterize()`: Sníží počet bitů použitých k reprezentaci každého pixelu na náhodnou hodnotu mezi 4 a 8.
    - `ShearY()`: Aplikuje vertikální zkosení obrázku o náhodný faktor mezi -0,1 a 0,1.
  - `Normalize()`: Provede normalizaci obrázků podle zadaného průměru a standardní odchylky.
  - `Cutout()`: Autory implementovaná transformace, která náhodně vyřízne obdélníkové díry z obrázku (tzn. překryje část obrázku černým obdélníkem).

Potenciálně kontraproduktivní mohou být funkce `RandomResizedCrop()`, `RandomCrop()` a `Cutout()`. Pokud v rámci těchto transformací dojde k zakrytí či oříznutí části obrázku obsahující informace o chybě, může dojít ke ztrátě informace o náležitosti ke správné třídě, jak je znázorněno na obrázcích 8.1 a 8.2.



Obrázek 8.1: Příklad obrázku s chybou, ve kterém by po náhodném ořezu mohla být ztracena informace o chybě.

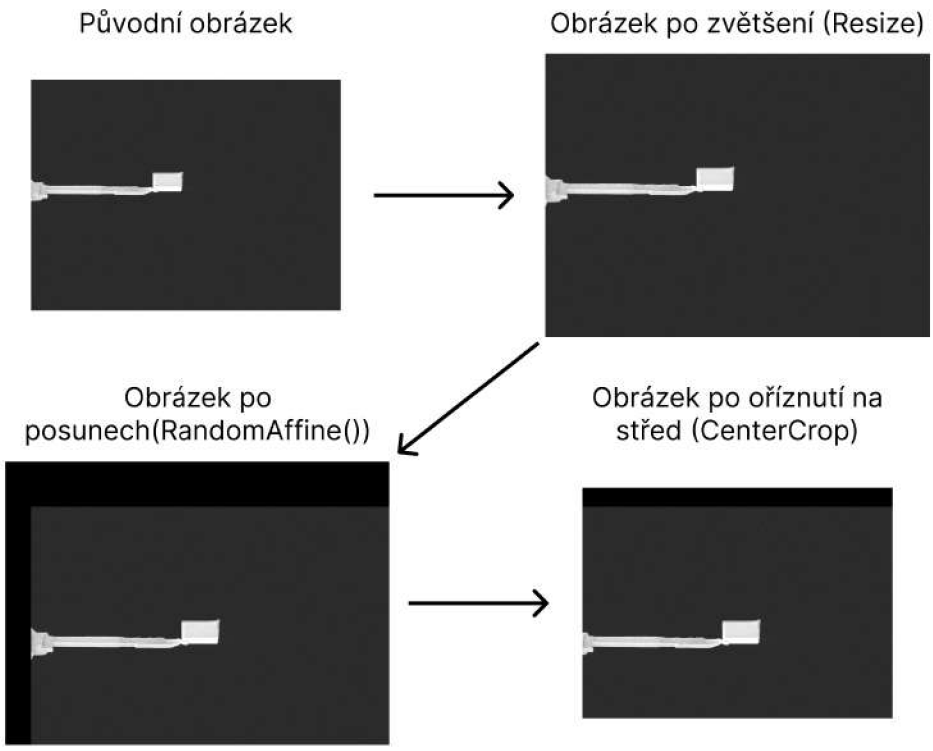


Obrázek 8.2: Příklad obrázku s chybou, ve kterém by po náhodném překrytí (Cutout) obrázky mohla být ztracena informace o chybě.

Z tohoto důvodu jsem navrhl vlastní sadu transformací, která v pretextové učení fázi nahrazuje transformaci `RandomResizedCrop()` a v následujících fázích nahrazuje transformace `RandomResizedCrop()` a `Cutout()`:

- `Resize()`: Změní velikost obrázku na zadanou velikost.
- `RendomAffine()`: Aplikuje afinní transformace na obrázek, specificky provede náhodný posun v horizontálním a vertikálním směru.
- `CenterCrop()`: Ořízne obrázek na centrální část obrázku o zadané velikosti.

Transformace `Resize()` nejdříve o něco zvětší rozměry obrázku, po čemž transformace `RendomAffine()` provede náhodný posun v horizontálním a vertikálním směru, jelikož obrázky patřící do stejné třídy jsou v datasetech TFS oproti sobě v těchto směrech často posunuté. Tím je však do obrázku zaveden černý pruh podél dvou stran a proto na závěr dojde pomocí transformace `CenterCrop()` k oříznutí na původní velikost obrázku, čímž se minimalizuje prostor zabíraný těmito černými pruhy, jak znázorňuje obrázek 8.3.



Obrázek 8.3: Příklad posloupnosti mnou navržených transformací a jejich efektů na transformovaný obrázek.

Dále jsem se rozhodl vyzkoušet dva druhy nahrazení původních transformací. V jedné variantě budou mými transformacemi nahrazeny výše zmíněné rizikové transformace ve všech fázích učení modelu SCAN a ve druhé pouze v pretextové fázi.

8.1.2 Vstupní rozlišení obrázků

Prvních několik měsíců jsem měl ve firmě TFS přístup k přístroji s grafickou kartou řady GeForce GTX 1060 s 10 GB paměti. Z toho důvodu jsem musel snížit rozlišení vstupních obrázků na 100x100 pixelů. Na závěr času vymezeného pro tuto práci jsem ale dostal přístup k přístroji s moderní grafickou kartou GeForce GTX 3090 s 24 GB paměti, proto jsem se rozhodl ověřit vliv rozlišení vstupních obrázků na výslednou úspěšnost modelu. Do závěrečného porovnání na konci této kapitoly jsem vybral rozlišení 128x128 a 256x256 pixelů.

8.1.3 Počet epoch při trénování

Dalším parametrem který jsem se rozhodl otestovat je počet epoch trénování modelu. Jelikož datasety z TFS obsahují podstatně méně obrázků, než dataset CIFAR10, rozhodl jsem se změřit vliv zdvojnásobení počtů epoch, které autoři používali při trénování, na úspěšnost modelu. U pretextové fáze tedy z 500 na 1000 epoch, u fáze Scan ze 100 na 200 epoch a u fáze Selflabel ze 200 na 400 epoch.

8.1.4 Fixní a dynamicky nastavovaný práh pro výběr důvěryhodných vzorků

Ve fázi Selflabel učení modelu SCAN dochází k výběru důvěryhodných vzorků na základě jistoty přiřazení k dané třídě získané z výstupu fáze Scan. Autoři tento práh nastavili fixně na pravděpodobnost náležitosti 0.99, jenže této hranice ve většině případů nedosáhne po fázi Scan žádný vzorek z testovaných datasetů TFS. Proto jsem upravil tuto fázi učení tak, aby umožňovala nastavení této hranice na nejvyšší dosaženou pravděpodobnost po fázi Scan.

8.2 Testování různých nastavení

Testování různých nastavení modelu SCAN spočívá v opakovaném spouštění trénování s různými kombinacemi hodnot výše uvedených parametrů. Trénování i měření úspěšnosti probíhalo vždy na celém datasetu obrázků, jelikož v praxi shlukování bude síť trénována na stejné skupině obrázků, kterou bude po konci trénování rozdělovat do shluků.

Spuštěním trénování se všemi čtyřiadvaceti kombinacemi výše zmíněných parametrů jsem získal tabulku různých nastavení a jim odpovídajících hodnot metrik Přesnost, ARI a NMI. Tuto tabulku jsem analyzoval s cílem zjistit, jaké hodnoty kterých parametrů měly největší vliv na výslednou úspěšnost. Kompletní získaná data jsou k nahlédnutí v příloze A.1.

Vzniklá tabulka tedy obsahuje následující závislé a nezávislé proměnné:

- Závislé proměnné
  - Přesnost
  - ARI
  - NMI
- Nezávislé proměnné
  - Transformační sada - nabývá hodnot `Originální transformace` pro trénování s nezměněnými transformacemi, `Mé transformace` pro trénování s použitím mnou definované transformační sady ve všech fázích učení a `Kombinované transformace` pro trénování s použitím mnou definované sady transformací pouze v první (pretextové) fázi.
  - Rozlišení - nabývá hodnot `128x128` a `256x256`.
  - Počet epoch - nabývá hodnot `Původní` pro publikované počty epoch a `Dvojnásobný` pro dvojnásobné počty epoch.
  - Práh - nabývá hodnot `Fixní práh` pro prahovou hodnotu důvěryhodnosti 0.99 a `Dynamický práh` pro dynamické nastavování hodnoty prahu na nejvyšší dosaženou hodnotu po konci fázi Scan.

Hodnota proměnné Přesnost může být zavádějící, metrika ARI je vhodnější pro vyvážené datasety a NMI pro nevyvážené (viz 5.4). Dataset Needle Attachment Small je s 52 vzorky v každé třídě vyvážený, proto jsem se rozhodl pro analýzu použít pouze hodnoty proměnné ARI.

8.2.1 Porovnání vlivu parametrů na výsledky modelu SCAN

Za účelem zjištění, jaké hodnoty kterých parametrů měly největší vliv na hodnotu metriky ARI by bylo ideální provést analýzu pomocí metody One-Way ANOVA [32] či jiných statistických metod jako například matice korelace [20]. Jelikož jsou však nezávislé proměnné kategorického typu a není důvod předpokládat, že by závislá intervalová proměnná spadala do Normálního rozložení pravděpodobnosti, nelze tyto tradiční metody pro analýzu dat použít.

Namísto toho jsem se tedy rozhodl pro jednoduché porovnání průměrné hodnoty ARI získané zafixováním hodnoty jedné z nezávislých proměnných a vypočtením průměru ARI ze všech běhů trénování s danou hodnotou parametru. Výsledky jsem zanesl do následujících tabulek pro každou fázi trénování zvlášť.

Tabulka 8.1: Výsledky po pretextové fázi učení

Proměnná	Hodnota	Průměr ARI
Rozlišení	128x128	0.543
	256x256	0.411
Transformační sada	Originální transformace	0.548
	Mé transformace	0.289
Počet epoch	Původní	0.446
	Dvojnásobný	0.508

Z tabulky výsledků po pretextové fázi učení 8.1 byla vynechána proměnná `Práh`, jelikož na tuto fázi nemá žádný vliv. Dále byla vynechána hodnota `Kombinované transformace` proměnné `Transformační sada`, jelikož v této fázi učení aplikuje stejné transformace jako sada `Mé transformace`.

Tabulka 8.2: Výsledky po fázi učení Scan

Proměnná	Hodnota	Průměr ARI
Rozlišení	128x128	0.566
	256x256	0.503
Transformační sada	Originální transformace	0.556
	Mé transformace	0.510
	Kombinované transformace	0.537
Počet epoch	Původní	0.529
	Dvojnásobný	0.539

Z tabulky výsledků po fázi učení Scan 8.2 byla vynechána proměnná `Práh`, jelikož na tuto fázi nemá žádný vliv.

Výsledky ve fázi Selflabel v tabulce 8.3 dosahují celkově výrazně nižších průměrů, jelikož pro hodnotu `Fixní práh` proměnné `Práh` je v případě, že ani jeden vzorek nedosáhne prahové hranice důvěry (tedy pravděpodobnosti náležitosti ke třídě 0.99) celá fáze učení přeskočena s nulovou výslednou hodnotou ARI. U dynamického hledání prahové hodnoty pak výsledky často degradovaly do jediného shluku, čemuž metrika ARI také přiřazuje nulovou hodnotu.

Tabulka 8.3: Výsledky po fázi učení Selflabel

Proměnná	Hodnota	Průměr ARI
Rozlišení	128x128	0.039
	256x256	0
Transformační sada	Originální transformace	0
	Mé transformace	0
	Kombinované transformace	0.059
Počet epoch	Původní	0
	Dvojnásobný	0.039
Práh	Fixní práh	0.039
	Dynamický práh	0

Analýza příčin špatných výsledků fáze Selflabel s dynamicky nastaveným prahem důvěryhodnosti

Špatná úspěšnost fáze selflabel s dynamicky nastaveným prahem byla překvapující - i když ve většině běhů trénování došlo ke snížení prahu důvěryhodnosti pod 0.9 (oproti publikovanému prahu 0.99), neočekával jsem až takto špatné výsledky. Po podrobnějším prostudování běhů trénování jsem došel k závěru, že stejně špatný vliv jako nízké hodnoty jistoty po fázi Scan má i nízký počet vzorků, přesahující stanovenou hranici důvěryhodnosti. Důvodem k tomuto závěru je fakt, že i při trénování, ve kterém dosáhl malý počet vzorků fixní prahové hodnoty důvěryhodnosti 0.99 nakonec výsledné shlukování zdegradovalo do jediného shluku.

Dynamické snížení prahu na nižší hodnotu, které ale opět dosáhne jen pár vzorků tedy problém neřeší, jelikož i tato úprava má za následek degradaci k jedinému shluku. Proto jsem se rozhodl provést experiment s fixně nastavenou hranicí důvěryhodnosti na 0.8, ale ani ten nedosáhl úspěšnosti, které dosáhlo samotné trénování po fázi Scan. Rozhodl jsem se tedy nechat nastavený práh důvěryhodnosti na fixní hodnotu 0.99 s tím, že na většině datasetů z TFS bude tento krok trénování pravděpodobně přeskočen.

8.3 Vyhodnocení

Na výsledcích po všech fázích učení lze pozorovat výrazně lepší úspěšnosti u proměnné `Počet epoch` pro hodnotu `Dvojnásobný`, což je pochopitelné, jelikož trénování i měření úspěšnosti bylo prováděno na stejných obrázcích.

Znatelně lepších výsledků dosahuje i nastavení parametru `Rozlišení` na hodnotu `128x128`. To je překvapivé a poněkud neintuitivní zjištění, ale možným vysvětlením by mohl být vliv malého poměru velikosti diskriminujícího objektu v obrázcích z první 6.1 a druhé 6.2 třídy vůči celkové velikosti obrázku. To je stav který negativně ovlivňuje výsledky klasifikačních sítí [35] a dá se předpokládat stejný negativní vliv i na model SCAN. Je možné, že tato negativní vlastnost obrázků měla větší efekt při rozlišení `256x256` než při rozlišení čtyřnásobně menším (`128x128`).

Mnou navržené sady transformací při testování nedosáhly vyšších hodnot metriky ARI než původní transformace publikované s modelem SCAN. Jedinou výjimku tvoří fáze selflabel, ale podrobnější analýze výsledků v této fázi učení ukazuje, že za mírně lepší než nulovou hodnotou ARI pro sadu transformací `Kombinované transformace` stojí jediný úspěšný výsledek trénování. Jelikož byl model trénován se sadou `Kombinované transformace` celkem osmkrát a výše zmíněný výsledek se vyskytl jen jednou, přiřazuji jeho úspěch spíše náhodné inicializaci shlukovací hlavy ve fázi Scan, než efektu transformací.

Dynamické hledání nižší prahové jistoty pro označení obrázku za důvěryhodný ve fázi Selflabel nastavením parametru `Práh` na `Dynamický práh` také nevedlo na úspěšné trénování. Pravděpodobným důvodem je zanesení chybně označených obrázků mezi důvěryhodné a nízký počet vzorků dosahující snížené hodnoty prahu po fázi Scan, což má za následek degradaci shlukování do jediného shluku.

Na základě těchto informací jsem zvolil následující finální nastavení modelu SCAN:

Tabulka 8.4: Finální nastavení modelu SCAN

Proměnná	Hodnota
Rozlišení	128x128
Transformační sada	Originální transformace
Počet epoch	Dvojnásobný
Práh	Fixní práh

Kapitola 9

Porovnání různých shlukovacích metod na datasetech z TFS

Nastavením experimentů pro porovnání úspěšnosti modelu SCAN s metodami pro shlukování podle metadat jsem se snažil co nejblíže přiblížit reálným případům použití softwaru ve firmě TFS. V této kapitole popíši výsledky porovnání na datasetech Needle Attachment Large, Needle Attachment Small a Welding Manipulator, přičemž každý dataset odpovídá modelovému případu použití.

Nastavení pro modelu scan je popsané v tabulce 8.4 a pro všechny datasety stejné. Mezi experimenty se ale mění batch-size [1], která je nastavena na 128, což je nejvyšší možný počet obrázků s rozlišením 128 na 128 pixelů zároveň s modelem uložitelných do paměti GPU o velikosti 24 GB. Dataset Welding Manipulator 6.3.2 však obsahuje pouze 47 obrázků, proto jsem pro něj nastavil batch size na 47.

9.1 Needle Attachment Large

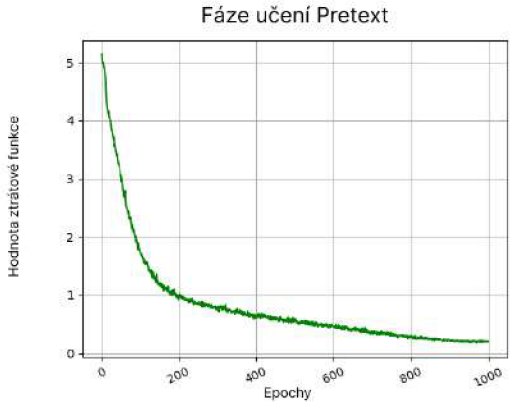
Needle Attachment Large 6.3.1 čítá 3040 obrázků z 233 různých kroků mikroskopu, které lze rozdělit tří tříd. Třídy jsou dostatečně vizuálně odlišné a odborníci z TFS předpokládají, že by se mezi sebou mohly lišit i v metadatech, i když ta se liší hlavně mezi různými kroky práce mikroskopu. Proto je otázka zda shlukovací metody dokáží 233 podshluků rozdělit správně na tři výsledné shluky.

9.1.1 Průběh trénování

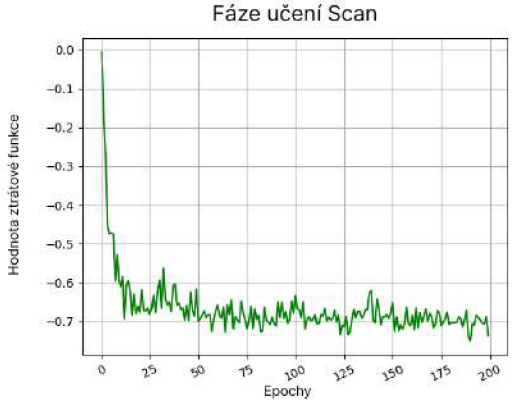
Grafy 9.1, 9.2 a 9.3 vykreslují pokles ztrátových funkcí v průběhu jednotlivých fází trénování modelu SCAN.

Z grafů učení ve fázi Pretext a Scan lze vyčíst, že trénování probíhalo očekávaným způsobem a ztrátová funkce byla až na mírné korekce stabilně snižována. Po fázi Scan dokonce několik vzorků dosáhlo hranice důvěryhodnosti 0.99 a učení pokračovalo fází Selflabel. Jednalo se ale pouze o jednotky z celkového počtu 3040 obrázků, což bylo velmi pravděpodobně příčinou degradace shlukování do jednoho shluku a praktického konce učení zhruba po 100 epochách.

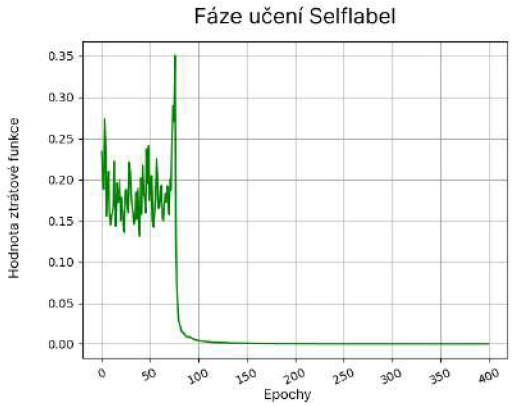
[1] Batch size udává počet obrázků, které projdou neuronovou sítí než dojde k úpravě vah sítě.



Obrázek 9.1: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN v pretextové fázi na datasetu Needle Attachment Large.



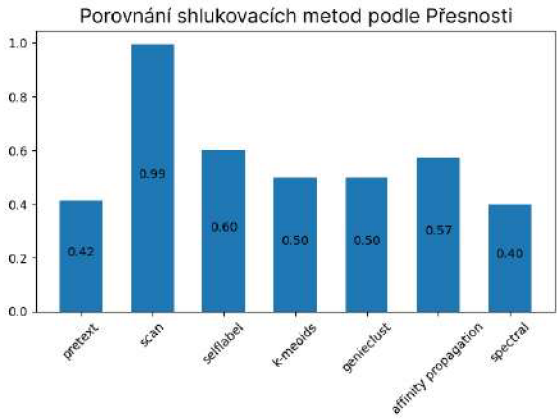
Obrázek 9.2: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN ve fázi Scan na datasetu Needle Attachment Large.



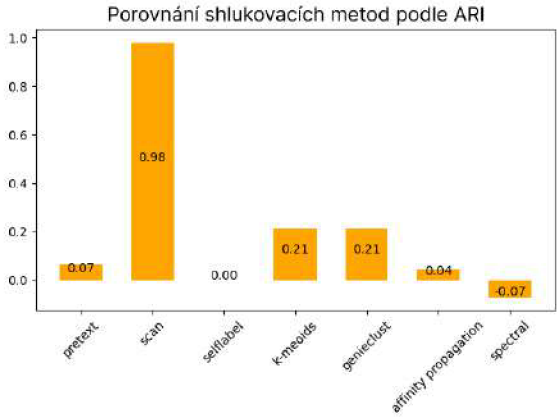
Obrázek 9.3: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN ve fázi Selflabel na datasetu Needle Attachment Large.

9.1.2 Vyhodnocení výsledků různých metod pro shlukování

Následující grafy 9.4, 9.5 a 9.6 porovnávají výsledky shlukování na datasetu Needle Attachment Large všech vybraných metod podle vybraných metrik úspěšnosti.

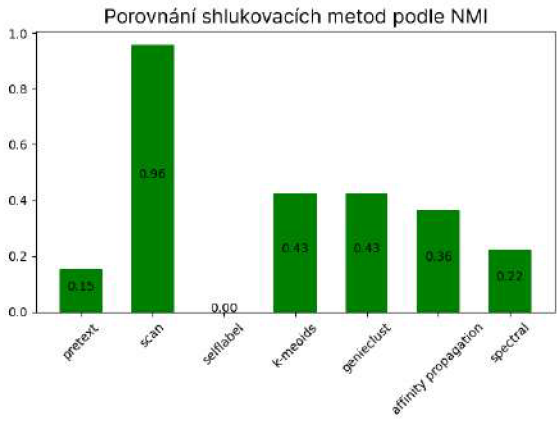


Obrázek 9.4: Graf porovnávající úspěšnost shlukovacích metod na datasetu Needle Attachment Small podle metriky Přesnost.



Obrázek 9.5: Graf porovnávající úspěšnost shlukovacích metod na datasetu Needle Attachment Small podle metriky ARI.

Na tomto datasetu dosáhl nejvyšší úspěšnosti model SCAN po fázi učení Scan. Po fázi Selflabel zdegradovalo shlukování do jednoho shluku a metody pro shlukování podle metadat dosáhly horších výsledků, jelikož se vybrané třídy mezi sebou v metadatech dostatečně jasně nelišily (metadata byla pravděpodobně odlišná mezi jednotlivými kroky a ne až tak mezi třídami).



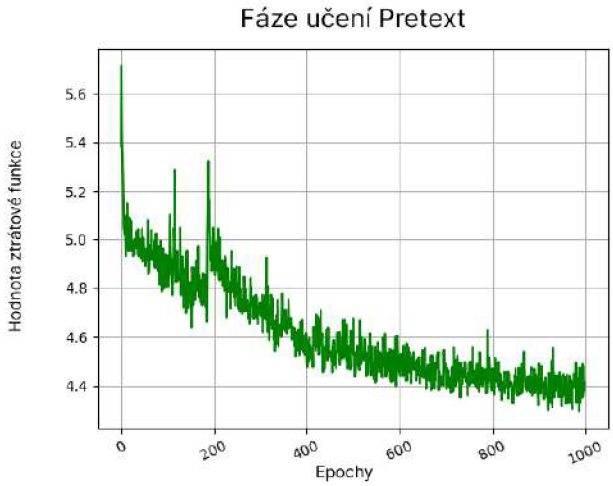
Obrázek 9.6: Graf porovnávající úspěšnost shlukovacích metod na datasetu Needle Attachment Small podle metriky NMI.

9.2 Needle Attachment Small

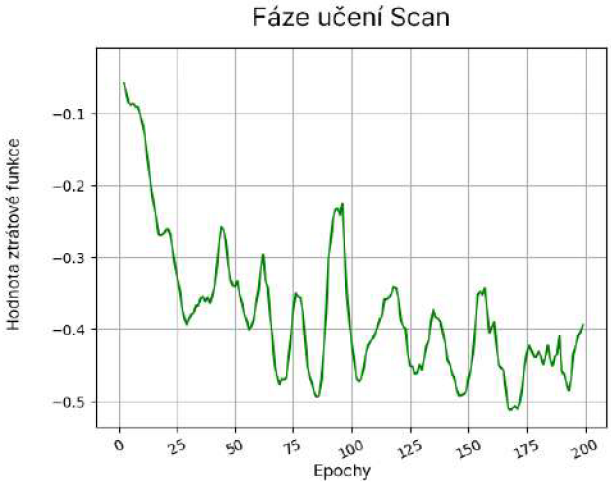
Needle Attachment Small 6.3.1 obsahuje tři třídy vizuálně odlišných obrázků ze tří různých kroků práce mikroskopu. Jedná se o nejjednodušší problém pro shlukování, jelikož třídy obrázků jsou vyvážené (každá třída obsahuje 52 obrázků), jasně vizuálně odlišné a pořízené v různých krocích práce mikroskopu (dá se tak předpokládat, že obsahují i různá metadata). Hlavní nevýhodou datasetu je pravděpodobně jeho malá velikost, jelikož obsahuje celkem jen 156 obrázků. Shlukování i takto malých datasetů by však v praxi bylo častým úkolem.

9.2.1 Průběh trénování

Grafy 9.7 a 9.8 vykreslují pokles ztrátové funkce v průběhu jednotlivých fází trénování modelu SCAN.



Obrázek 9.7: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN v pretextové fázi na datasetu Needle Attachment Small.

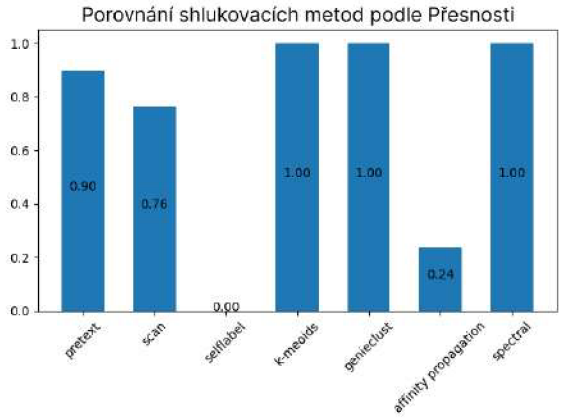


Obrázek 9.8: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN ve fázi Scan na datasetu Needle Attachment Small.

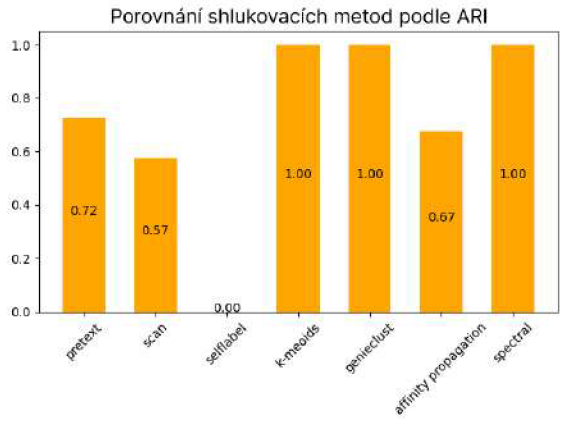
Z obou grafů lze vyčíst, že trénování probíhalo smysluplně a ztrátová funkce byla i přes výkyvy celkově snižována. Po fázi Scan však žádný ze vzorků nedosáhl požadované hranice jistoty 0.99 a proto byla fáze Selflabel vynechána.

9.2.2 Vyhodnocení výsledků různých metod pro shlukování

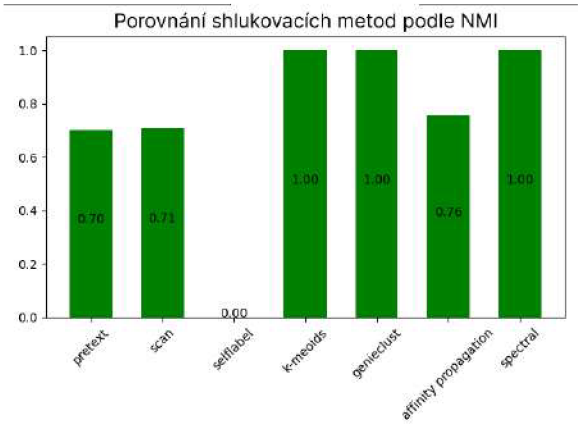
Následující grafy 9.9, 9.10 a 9.11 porovnávají výsledky všech vybraných metod pro shlukování na datasetu Needle Attachment Small podle vybraných metrik úspěšnosti.



Obrázek 9.9: Graf porovnávající úspěšnost shlukovacích metod na datasetu Needle Attachment Small podle metriky Přesnost.



Obrázek 9.10: Graf porovnávající úspěšnost shlukovacích metod na datasetu Needle Attachment Small podle metriky ARI.



Obrázek 9.11: Graf porovnávající úspěšnost shlukovacích metod na datasetu Needle Attachment Small podle metriky NMI.

Z těchto výsledků je patrné, že předpoklad o schopnosti rozlišovat mezi jednotlivými kroky podle metadat byl správný. Pro shlukování obrázků z různých kroků algoritmu je zde jasně nejvhodnější shlukování podle metadat, které je zároveň méně časově náročné, než trénování modelu SCAN. Nízkou úspěšnost algoritmu `Affinity Clustering` lze vysvětlit tím, že jako jediný z algoritmů shlukujících podle metadat nezná přesný počet shluků, do kterých má datové body rozdělit. Nízká úspěšnost modelu SCAN je pak pravděpodobně způsobená nízkým počtem obrázků v datasetu, jelikož je počet obrázků prakticky jediným rozdílem oproti předchozímu experimentu 9.1, ve kterém dosáhl model skvělé úspěšnosti.

Pro výsledné použití na podobné typy shlukování v praxi jsem zvolil algoritmus GenieClust, jelikož autoři prezentují rychlost srovnatelnou s algoritmem K-Means [15], jehož časová složit je nižší než u algoritmu K-Medoids [23] a výrazně nižší než časová složitost algoritmu Spectral Clustering [28].

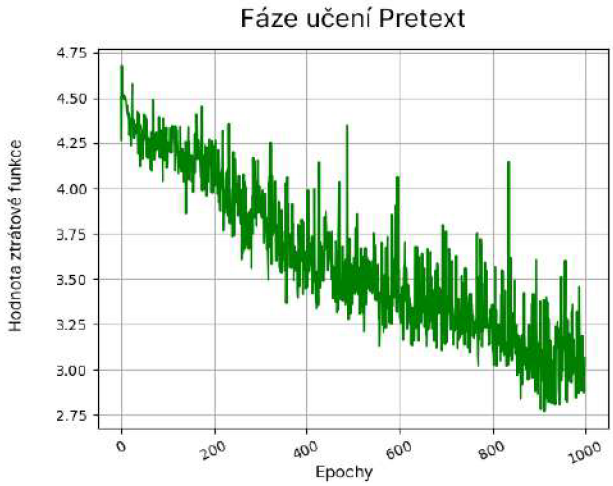
9.3 Welding Manipulator

Dataset Welding Manipulator obsahuje obrázky s chybou a bez chyby z jednoho kroku práce mikroskopu. Tento modelový případ představuje největší výzvu pro shlukování. Jedná se o rozdělení obrázků z identického kroku práce algoritmu na obrázky zobrazující korektní stav a obrázky zobrazující chybu, ke které došlo v rámci automatizované práce mikroskopu. V případě datasetu Welding Manipulator se jedná o obrázky pořízené při transportu lamely na transportní jehle z místa výřezu na místo pro ztenčení. Korektní obrázek zde obsahuje transportní jehlu s lamelou přivařenou na její špičce, chybný obrázek zachycuje samotnou jehlu bez lamely, jelikož lamela v průběhu transportu z jehly odpadla.

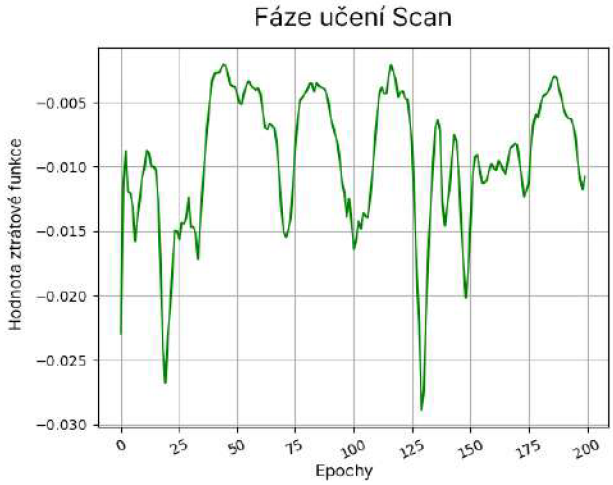
Dataset se sestává pouze ze 47 obrázků a navíc jsou počty korektních a chybných obrázků silně nevyvážené - korektních obrázků je 40 a chybných pouze 7, což ještě zvyšuje obtížnost správného shlukování.

9.3.1 Průběh trénování

Následující grafy 9.12 a 9.13 vykreslují pokles ztrátové funkce v průběhu jednotlivých fází trénování modelu SCAN.



Obrázek 9.12: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN v pretextové fázi na datasetu Welding Manipulator.



Obrázek 9.13: Graf vykresluje pokles ztrátové funkce v průběhu trénování modelu SCAN ve fázi Scan na datasetu Welding Manipulator.

Z grafů lze vyčíst, že trénování v pretextové fázi probíhalo s většími výkyvy, ale v průměru dosahovalo stabilního snižování ztrátové funkce. Oproti tomu učení ve fázi Scan nedosáhlo žádného významného snížení ztrátové funkce, což bylo pravděpodobně následkem toho, že nejbližší sousedé vzorků získaní po pretextové fázi nespadali ve potřebné většině případů do stejné třídy jako vzorky samotné. Po Scan fázi žádný ze vzorků nedosáhl požadované hranice jistoty 0.99 a proto byla fáze Selflabel vynechána.

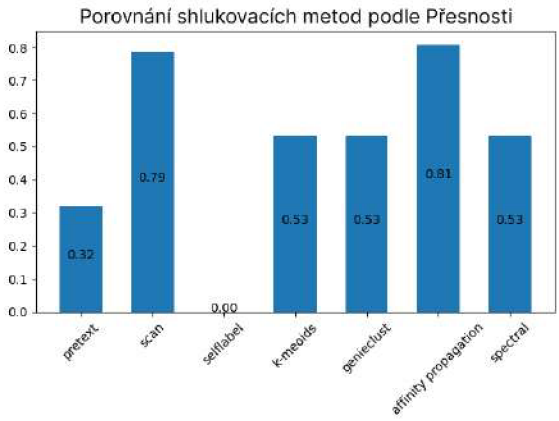
9.3.2 Vyhodnocení výsledků různých metod pro shlukování

Následující grafy 9.14, 9.15 a 9.16 porovnávají výsledky všech vybraných metod pro shlukování na datasetu Welding Manipulator podle vybraných metrik úspěšnosti.

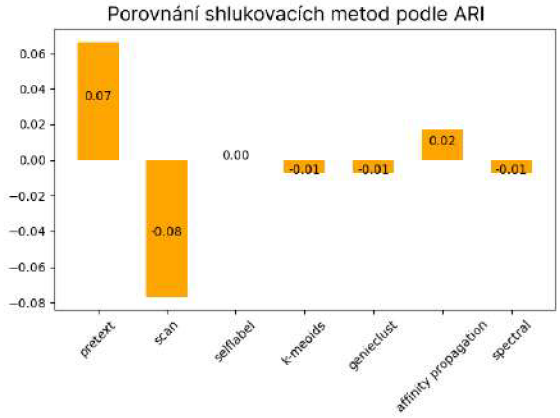
Obavy ohledně přílišné obtížnosti rozdělení takto podobných obrázků na dva silně nevyvážené shluky se potvrdily a všechny metody rozdělily obrázky do shluků prakticky náhodně. Vysoké hodnoty metriky Přesnost jsou zde způsobeny nevyvážeností datasetu - například přiřazení všech obrázků do jediného shluku by dosáhlo přesnosti 0.85 (jelikož 40 ze 47 obrázků by bylo klasifikováno jako správně přiřazených). Metriky ARI a NMI ale jasně ukazují, že výsledné shluky nesdílí žádné reálné informace s rozdělením do ground-truth tříd.

Z tohoto experimentu vyplývá, že tvorba datasetů pro trénování klasifikačních neuronových sítí za účelem odhalení chyb z obrazového výstupu automatizované práce elektronových mikroskopů bude muset nadále probíhat ručně.

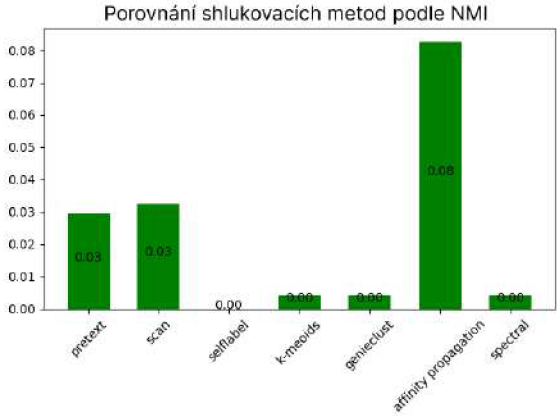
Ke zvážení je ještě implementace modelu SPICE, který na datasetech pro porovnávání modelů dosahuje v době psaní této práce SOTA výsledků, toto rozhodnutí už ale zůstává na firmě TFS, jelikož s sebou nese nutnost investice do stavby stroje se čtyřmi jednotkami GPU.



Obrázek 9.14: Graf porovnávající úspěšnost shlukovacích metod na datasetu Welding Manipulator podle metriky Přesnost.



Obrázek 9.15: Graf porovnávající úspěšnost shlukovacích metod na datasetu Welding Manipulator podle metriky ARI.



Obrázek 9.16: Graf porovnávající úspěšnost shlukovacích metod na datasetu Welding Manipulator podle metriky NMI.

Kapitola 10

Závěr

Cílem práce bylo implementovat systém pro shlukování obrázků z elektronových mikroskopů, s tímto systémem experimentovat a porovnat výsledky dosažené shlukováním podle vizuální podobnosti s výsledky shlukování podle metadat. Tyto cíle práce jsem splnil a navíc vyhodnotil použitelnost zde popsaných metod shlukování na praktické problémy firmy Thermo Fisher Scientific.

Pro naplnění výše uvedených cílů bylo třeba se seznámit s metodami pro shlukování obrazových dat s pomocí i bez pomoci metadat. Tento požadavek jsem splnil nastudováním moderních shlukovacích metod a modelů, které jsem mezi sebou porovnal a vybral ty nejvhodnější pro shlukování obrázků z elektronové mikroskopie. Následně jsem implementoval systém pro porovnání shlukování pomocí všech vybraných metod. Se systémem jsem experimentoval a porovnal výsledky pomocí několika metrik pro měření úspěšnosti shlukování a vyvodil z nich závěry ohledně použitelnosti jednotlivých metod pro účely firmy TFS v praxi.

Nad rámec zadání práce jsem před samotným porovnáním provedl experimenty za účelem nalezení optimálního nastavení parametrů modelu SCAN a zlepšení jeho výsledků na datasetech z TFS.

Závěrem práce je vyhodnocení použitelnosti shlukovacích metod pro typické případy použití v praxi. Pro automatické rozdělování sad obrázků do tříd odpovídajících různým krokům práce mikroskopu se ukázaly být nejlepší metody K-Means, GenieClust a Spectral Clustering. Všechny tyto metody dosáhly na datasetu odpovídajícímu tomuto případu použití 100% úspěšnosti. Metoda GenieClust ale provede shlukování nejrychleji a proto jsem doporučil právě ji pro použití v praxi. Pro rozdělování řádově tisíců obrázků z většího počtu kroků mikroskopu do tří tříd už ale výsledkům shlukování dominoval model SCAN, který po fázi učení Scan dosáhl 99% úspěšnosti. Použitelnost v této práci popsaných metod pro shlukování na rozdělování obrázků z různých kroků práce mikroskopu se tedy potvrdila.

Naopak rozdělování obrázků na obrázky korektní a obrázky zachycující chybný stav automatizované práce mikroskopu se ukázalo být pro metody shlukování bez učitele příliš náročným problémem. Tyto datasety v praxi čítají jen desítky až stovky obrázků, což je pro učení bez učitele velmi malý počet. Rozdíl mezi chybným a korektním obrázkem se navíc v elektronové mikroskopii často ukrývá v jemných detailech, které by byly těžko rozlišitelné i pro netrénovaného člověka. To jsou pravděpodobně hlavní příčiny, proč se žádné z metod pro shlukování nepodařilo dosáhnout použitelné úspěšnosti. Výsledkem experimentů se shlukováním podle metadat je ale i hodnotné zjištění, že informace o chybném stavu se neukrývá ve specifickém nastavení mikroskopu odlišitelném shlukovacími algoritmy.

Na práci by bylo možné navázat implementací modelu SPICE, který by jakožto současný SOTA model mohl dosáhnout lepších výsledků než mnou použitý model SCAN. S tím se však pojí nemalá investice do sestavení přístroje se čtyřmi výkonnými jednotkami GPU, které model pro trénování vyžaduje a proto zůstává rozhodnutí o tomto pokračování na firmě TFS jakožto zadavateli práce.

Literatura

[1] *Image Clustering on CIFAR-10* [online]. [cit. 2023-01-14]. Dostupné z: https://paperswithcode.com/sota/image-clustering-on-cifar-10?metric=Accuracy.

[2] Ali, A., Bin Faheem, Z., Waseem, M., Draz, U., Safdar, Z. et al. Systematic Review: A State of Art ML Based Clustering Algorithms for Data Mining. In: *2020 IEEE 23rd International Multitopic Conference (INMIC)*. 2020, s. 1–6. DOI: 10.1109/INMIC50486.2020.9318060.

[3] Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A. et al. *MixMatch: A Holistic Approach to Semi-Supervised Learning*. 2019.

[4] Cai, S., Qiu, L., Chen, X., Zhang, Q. a Chen, L. *Semantic-enhanced Image Clustering*. arXiv, 2022. DOI: 10.48550/ARXIV.2208.09849. Dostupné z: https://arxiv.org/abs/2208.09849.

[5] Chauhan, N. S. *DBSCAN Clustering Algorithm in Machine Learning* [online], 4. dubna 2022 [cit. 2023-01-16]. Dostupné z: https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html.

[6] Chen, T., Kornblith, S., Norouzi, M. a Hinton, G. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020.

[7] community scikit. *Clustering* [online]. [cit. 2023-01-14]. Dostupné z: https://scikit-learn.org/stable/modules/clustering.html.

[8] community, S. *Clustering*. 2023. Dostupné z: https://scikit-learn.org/stable/modules/clustering.html.

[9] Contributors, T. *Torchvision.Transforms*. 2023. Dostupné z: https://pytorch.org/vision/0.9/transforms.html.

[10] Dridi, S. *Unsupervised Learning - A Systematic Literature Review*. Prosinec 2021. DOI: 10.13140/RG.2.2.16963.12323.

[11] Englesson, E. a Azizpour, H. *Consistency Regularization Can Improve Robustness to Label Noise*. 2021.

[12] Ester, M., Kriegel, H.-P., Sander, J. a Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In:. AAAI Press, 1996, s. 226–231. KDD'96.

[13] Frey, B. J. a Dueck, D. Clustering by Passing Messages Between Data Points. *Science*. 2007, sv. 315, s. 972 – 976.

[14] Fukunaga, K. a Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*. 1975, sv. 21, č. 1, s. 32–40. DOI: 10.1109/TIT.1975.1055330.

[15] Gagolewski, M. Genieclust: Fast and robust hierarchical clustering. *SoftwareX*. 2021, sv. 15, s. 100722. DOI: https://doi.org/10.1016/j.softx.2021.100722. ISSN 2352-7110. Dostupné z: https://www.sciencedirect.com/science/article/pii/S2352711021000649.

[16] Gansbeke, W. V., Vandenhende, S., Georgoulis, S., Proesmans, M. a Gool, L. V. Learning To Classify Images Without Labels. *CoRR*. 2020, abs/2005.12320. Dostupné z: https://arxiv.org/abs/2005.12320.

[17] Goodfellow, I., Bengio, Y. a Courville, A. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[18] Gordon Rodriguez, E., Loaiza Ganem, G., Pleiss, G. a Cunningham, J. P. *Uses and Abuses of the Cross-Entropy Loss: Case Studies in Modern Deep Learning*. 2020.

[19] Gower, J. C. A General Coefficient of Similarity and Some of Its Properties. *Biometrics*. JSTOR. dec 1971, sv. 27, č. 4, s. 857. DOI: 10.2307/2528823. Dostupné z: https://doi.org/10.2307/2528823.

[20] Hair, J. F. *Multivariate Data Analysis: An Overview*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. 904–907 s. ISBN 978-3-642-04898-2. Dostupné z: https://doi.org/10.1007/978-3-642-04898-2_395.

[21] Hubert, L. a Arabie, P. Comparing partitions. *Journal of Classification*. Dec 1985, sv. 2, č. 1, s. 193–218. DOI: 10.1007/BF01908075. ISSN 1432-1343. Dostupné z: https://doi.org/10.1007/BF01908075.

[22] Karypis, G., Han, E.-H. a Kumar, V. Chameleon: hierarchical clustering using dynamic modeling. *Computer*. 1999, sv. 32, č. 8, s. 68–75. DOI: 10.1109/2.781637.

[23] Kaufman, L. a Rousseeuw, P. *Finding Groups in Data: An Introduction To Cluster Analysis*. Leden 1990. ISBN 0-471-87876-6.

[24] Krizhevsky, A. *Cifar-10 and Cifar100 datasets* [online]. [cit. 2023-01-14]. Dostupné z: https://www.cs.toronto.edu/~kriz/cifar.html.

[25] Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images. *University of Toronto*. Květen 2012.

[26] Kuhn, H. W. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*. March 1955, sv. 2, 1–2, s. 83–97. DOI: 10.1002/nav.3800020109.

[27] Li, J., Socher, R. a Hoi, S. C. H. *DivideMix: Learning with Noisy Labels as Semi-supervised Learning*. 2020.

[28] Li, M., Lian, X.-C., Kwok, J. T. a Lu, B.-L. Time and space efficient spectral clustering via column sampling. In: *CVPR 2011*. 2011, s. 2297–2304. DOI: 10.1109/CVPR.2011.5995425.

[29] Morbieu, S. *Accuracy: from classification to clustering evaluation* [online]. [cit. 2023-03-07]. Dostupné z: https://smorbieu.gitlab.io/accuracy-from-classification-to-clustering-evaluation/.

[30] Ng, A., Jordan, M. a Weiss, Y. On Spectral Clustering: Analysis and an algorithm. In: Dietterich, T., Becker, S. a Ghahramani, Z., ed. *Advances in Neural Information Processing Systems*. MIT Press, 2001, sv. 14. Dostupné z: https://proceedings.neurips.cc/paper_files/paper/2001/file/801272ee79cfde7fa5960571fee36b9b-Paper.pdf.

[31] Niu, C., Shan, H. a Wang, G. *SPICE: Semantic Pseudo-labeling for Image Clustering*. arXiv, 2021. DOI: 10.48550/ARXIV.2103.09382. Dostupné z: https://arxiv.org/abs/2103.09382.

[32] Ostertagova, E. a Ostertag, O. Methodology and Application of One-way ANOVA. *American Journal of Mechanical Engineering*. Listopad 2013, sv. 1, s. 256–261. DOI: 10.12691/ajme-1-7-21.

[33] Park, S., Han, S., Kim, S., Kim, D., Park, S. et al. Improving Unsupervised Image Clustering With Robust Learning. In:. 2021.

[34] Patel, S., Sihmar, S. a Jatain, A. A study of hierarchical clustering algorithms. In: *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*. 2015, s. 537–541.

[35] Richter, M. L., Byttner, W., Krumnack, U., Wiedenroth, A., Schallner, L. et al. (Input) Size Matters for CNN Classifiers. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2021, s. 133–144. DOI: 10.1007/978-3-030-86340-1_11.

[36] Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R. et al. Clustering algorithms: A comparative approach. *PLOS ONE*. Public Library of Science. Leden 2019, sv. 14, č. 1, s. 1–34. DOI: 10.1371/journal.pone.0210236. Dostupné z: https://doi.org/10.1371/journal.pone.0210236.

[37] Romano, S., Vinh, N. X., Bailey, J. a Verspoor, K. *Adjusting for Chance Clustering Comparison Measures*. 2015.

[38] Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H. et al. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. a Lin, H., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020, sv. 33, s. 596–608. Dostupné z: https://proceedings.neurips.cc/paper/2020/file/06964dce9addb1c5cb5d6e3d9838f733-Paper.pdf.

[39] Steinbach, M., Karypis, G. a Kumar, V. A Comparison of Document Clustering Techniques. *Proceedings of the International KDD Workshop on Text Mining*. Červen 2000.

[40] Strehl, A. a Ghosh, J. Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*. Leden 2002, sv. 3, s. 583–617. DOI: 10.1162/153244303321897735.

[41] Van Gansbeke, W. *Unsupervised-Classification* [https://github.com/wvangansbeke/Unsupervised-Classification]. GitHub, 2023 [cit. 2022-11-10].

[42] Zhang, T., Ramakrishnan, R. a Livny, M. BIRCH: An Efficient Data Clustering Method for Very Large Databases. *SIGMOD Rec.* New York, NY, USA: Association for Computing Machinery. jun 1996, sv. 25, č. 2, s. 103–114. DOI: 10.1145/235968.233324. ISSN 0163-5808. Dostupné z: https://doi.org/10.1145/235968.233324.

Příloha A

Výsledky testování různých nastavení modelu SCAN

V této příloze jsou v tabulce A.1 uvedena data nasbíraná při testování různých nastavení modelu SCAN. Data jsou seřazena podle sloupců Fáze trénování a Přesnost. Tabulka samotná je příliš široká pro zobrazení všech sloupců na formátu A4 na výšku, proto ji naleznete na dalších stránkách, které jsou orientované na šířku.

Tabulka A.1: Tabulka výsledků testování různých nastavení modelu SCAN.

Rozlišení	Transformační sada	Počet epoch	Práh	Fáze trénování	Přesnost	ARI	NMI
256	Mé transformace	Dvojnásobný	Fixní práh	pretext	0.419354839	0.025259645	0.032840324
256	Mé transformace	Původní	Fixní práh	pretext	0.561290323	0.147674618	0.181989432
128	Originální transformace	Dvojnásobný	Dynamický práh	pretext	0.567741935	0.445711955	0.614611517
256	Mé transformace	Původní	Dynamický práh	pretext	0.574193548	0.244349127	0.252667026
128	Mé transformace	Původní	Fixní práh	pretext	0.593548387	0.293997721	0.371958385
256	Mé transformace	Dvojnásobný	Dynamický práh	pretext	0.6	0.196649527	0.199737277
256	Kombinované transformace	Původní	Dynamický práh	pretext	0.632258065	0.34131327	0.417826612
128	Mé transformace	Dvojnásobný	Dynamický práh	pretext	0.651612903	0.382740179	0.393480099
128	Originální transformace	Původní	Fixní práh	pretext	0.716129032	0.498076869	0.556493272
256	Originální transformace	Dvojnásobný	Dynamický práh	pretext	0.722580645	0.434930712	0.480496483
128	Mé transformace	Původní	Dynamický práh	pretext	0.735483871	0.521045504	0.598028082
128	Kombinované transformace	Původní	Fixní práh	pretext	0.761290323	0.522916148	0.58065304
128	Mé transformace	Dvojnásobný	Fixní práh	pretext	0.780645161	0.502997657	0.513803789
128	Originální transformace	Původní	Dynamický práh	pretext	0.780645161	0.52450749	0.556246551
256	Originální transformace	Původní	Fixní práh	pretext	0.780645161	0.527001965	0.563814336
256	Originální transformace	Původní	Dynamický práh	pretext	0.787096774	0.574098419	0.64377336
128	Kombinované transformace	Původní	Dynamický práh	pretext	0.8	0.564975739	0.602434791
256	Kombinované transformace	Původní	Fixní práh	pretext	0.8	0.592324444	0.675008615
256	Kombinované transformace	Dvojnásobný	Dynamický práh	pretext	0.819354839	0.57139664	0.597381201
256	Originální transformace	Dvojnásobný	Fixní práh	pretext	0.832258065	0.585281244	0.576471496
256	Kombinované transformace	Dvojnásobný	Fixní práh	pretext	0.883870968	0.68595919	0.635549891
128	Kombinované transformace	Dvojnásobný	Fixní práh	pretext	0.890322581	0.7219591	0.734747451
128	Kombinované transformace	Dvojnásobný	Dynamický práh	pretext	0.903225806	0.740169874	0.710851089
128	Originální transformace	Dvojnásobný	Fixní práh	pretext	0.929032258	0.796997478	0.752777408

Tabulka A.2: Pokračování tabulky výsledků testování různých nastavení modelu SCAN.

Rozlišení	Transformační sada	Počet epoch	Práh	Fáze trénování	Přesnost	ARI	NMI
256	Mé transformace	Původní	Dynamický práh	scan	0.64516129	0.241972977	0.351331204
128	Kombinované transformace	Dvojnásobný	Dynamický práh	scan	0.651612903	0.457929784	0.50681039
256	Mé transformace	Dvojnásobný	Dynamický práh	scan	0.651612903	0.50510572	0.571338978
128	Kombinované transformace	Původní	Fixní práh	scan	0.670967742	0.570637119	0.734777468
128	Mé transformace	Původní	Dynamický práh	scan	0.670967742	0.565781384	0.708804125
256	Mé transformace	Původní	Fixní práh	scan	0.670967742	0.553808863	0.669027707
256	Originální transformace	Původní	Fixní práh	scan	0.677419355	0.566108849	0.718740608
128	Originální transformace	Dvojnásobný	Dynamický práh	scan	0.683870968	0.530089919	0.622753028
256	Originální transformace	Dvojnásobný	Dynamický práh	scan	0.683870968	0.559663819	0.690723009
256	Kombinované transformace	Původní	Fixní práh	scan	0.690322581	0.540912834	0.641801748
256	Kombinované transformace	Původní	Dynamický práh	scan	0.709677419	0.532909367	0.624377042
256	Mé transformace	Dvojnásobný	Fixní práh	scan	0.722580645	0.332820627	0.40021809
256	Originální transformace	Původní	Dynamický práh	scan	0.729032258	0.513894839	0.591210513
128	Originální transformace	Původní	Dynamický práh	scan	0.748387097	0.564035645	0.682411291
128	Kombinované transformace	Původní	Dynamický práh	scan	0.761290323	0.538757933	0.608604733
128	Kombinované transformace	Dvojnásobný	Fixní práh	scan	0.767741935	0.543185167	0.610183462
128	Originální transformace	Dvojnásobný	Fixní práh	scan	0.767741935	0.547638004	0.617877234
256	Originální transformace	Dvojnásobný	Fixní práh	scan	0.767741935	0.57688808	0.709545633
256	Kombinované transformace	Dvojnásobný	Fixní práh	scan	0.787096774	0.556547416	0.602672234
128	Mé transformace	Původní	Fixní práh	scan	0.793548387	0.575473518	0.640214187
128	Originální transformace	Původní	Fixní práh	scan	0.793548387	0.589022551	0.682548354
256	Kombinované transformace	Dvojnásobný	Dynamický práh	scan	0.8	0.553750883	0.584102643
128	Mé transformace	Dvojnásobný	Dynamický práh	scan	0.838709677	0.639782332	0.709022019
128	Mé transformace	Dvojnásobný	Fixní práh	scan	0.858064516	0.667030726	0.714660539

Tabulka A.3: Pokračování tabulky výsledků testování různých nastavení modelu SCAN.

Rozlišení	Transformační sada	Počet epoch	Práh	Fáze trénování	Přesnost	ARI	NMI
128	Mé transformace	Dvojnásobný	Fixní práh	selflabel	0	0	0
128	Originální transformace	Dvojnásobný	Fixní práh	selflabel	0	0	0
128	Originální transformace	Dvojnásobný	Dynamický práh	selflabel	0	0	0
256	Kombinované transformace	Dvojnásobný	Fixní práh	selflabel	0	0	0
256	Originální transformace	Dvojnásobný	Fixní práh	selflabel	0	0	0
256	Originální transformace	Dvojnásobný	Dynamický práh	selflabel	0	0	0
128	Mé transformace	Původní	Fixní práh	selflabel	0	0	0
128	Originální transformace	Původní	Fixní práh	selflabel	0	0	0
128	Originální transformace	Původní	Dynamický práh	selflabel	0	0	0
256	Kombinované transformace	Původní	Fixní práh	selflabel	0	0	0
256	Kombinované transformace	Původní	Dynamický práh	selflabel	0	0	0
256	Mé transformace	Původní	Fixní práh	selflabel	0	0	0
256	Originální transformace	Původní	Fixní práh	selflabel	0	0	0
256	Originální transformace	Původní	Dynamický práh	selflabel	0	0	0
128	Kombinované transformace	Dvojnásobný	Dynamický práh	selflabel	0.335483871	0	0
128	Mé transformace	Dvojnásobný	Dynamický práh	selflabel	0.335483871	0	0
256	Kombinované transformace	Dvojnásobný	Dynamický práh	selflabel	0.335483871	0	0
256	Mé transformace	Dvojnásobný	Fixní práh	selflabel	0.335483871	0	0
256	Mé transformace	Dvojnásobný	Dynamický práh	selflabel	0.335483871	0	0
128	Kombinované transformace	Původní	Fixní práh	selflabel	0.335483871	0	0
128	Kombinované transformace	Původní	Dynamický práh	selflabel	0.335483871	0	0
128	Mé transformace	Původní	Dynamický práh	selflabel	0.335483871	0	0
256	Mé transformace	Původní	Dynamický práh	selflabel	0.335483871	0	0
128	Kombinované transformace	Dvojnásobný	Fixní práh	selflabel	0.658064516	0.468196104	0.53614852