Tiny Learning: Instance Selection Methods

Interim Report

Submitted by:

Muhammad Salman Malik, 20120791

Supervised by:

Dr Tomas Henrique Bode Maul

Submitted on: December 20, 2021

Table of Contents

1. Introduction

The rapid progress of technological advancement within the past few decades has led to an exponential growth in processing capabilities, which resulted in an explosion of data being created and collected at an enormous scale. Therefore, in this era of big data, more than ever, there is a need to make this abundance of data more tractable. While several approaches have been developed to tackle this issue of data reduction or compression, this report will focus mainly on instance selection techniques.

According to (García et al., 2015), instance selection (IS) is a task that consists of choosing a subset of the total available data to achieve the original purpose of the data mining (or machine learning) application as if the whole data had been used. In other words, it is the sampling of data into a subset containing ideally the best possible instances that has the same underlying distribution as the original dataset. At the same time, maintaining or even improving accuracy without any loss in performance when applying the same machine learning algorithm to both the original dataset and the subset. This often results in a trade-off that needs to be made between the *amount of reduction* and *prediction* or *classification quality* of the instance selection technique in question. (Chien-Hsing Chou et al., 2006).

The objective of scaling down data using Instance Selection methods is achieved in 2 ways:

by removing noisy or corrupt instances from training data

by removing superfluous or redundant instances from the training set

IS techniques that fit into the first category can be considered as competence enhancement techniques as they, while those that fit into the latter category may be referred to as competence preserving techniques (Cunningham & Delany, 2021). A more detailed taxonomy will be described in the Background Review: Taxonomy section.

Therefore, for a lot of data mining tasks, especially those involving large datasets, instance selection can prove to be a pivotal pre-processing step. After performing IS techniques on a training dataset, the resulting compressed dataset not only improves training speed, but also during prediction when using a particular algorithm (Blachnik, 2015). The speed gains in the prediction phase depends on the type of ML method used. For algorithms which do not have a fixed internal representation, this speed gain can be a result of fewer support vectors required in the case of SVMs, shallower trees when dealing with decision trees, or even lower number of reference vectors in kNN.

While the volume of data is progressively increasing each year, the state-of-the-art ML methods have not been keeping up with the same pace of growth. Furthermore, the computational complexity of many of these methods grow faster than the size of the datasets.

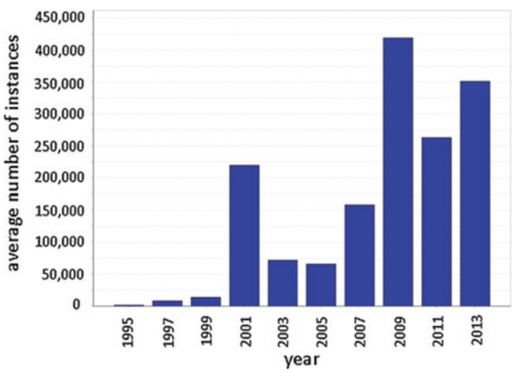


Figure 1. (Asuncion, A., Newman, D., 2007): the average size of datasets submitted to the UCI repository per year.

The main aim of this research is to innovate on existing instance selection or prototype selection approaches in the context of data minimization/compression in the domain of deep learning. In addition, the most popular traditional algorithms from the literature will be explored and compared along with more recent state-of-the-art IS techniques.

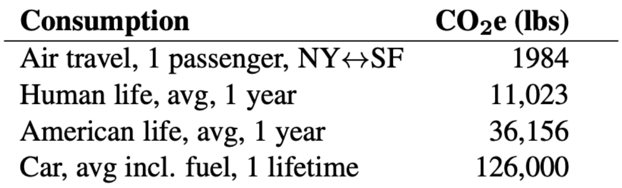
Gaps

There is a far larger body of work in the literature focusing on feature selection compared to instance selection, however both are quite significant data reduction or minimization techniques and require equal consideration. Since instance selection is closely related to feature selection, a combination of both methods seems quite promising.

Another potential gap that I identified was ensembles of IS. Despite the popularity of ensemble methods in developing predictive models, ensembles of instance selection methods fail to attract many researchers and are largely neglected (Blachnik, 2019). Therefore adapting ensembles to IS techniques could be an interesting gap to fill.

2. Motivation

Every 3.4 months, the average computational resource needs for state-of-the-art deep learning models has doubled, which has resulted in an astounding 300,000 fold increase from 2012 to 2018 (Strubell et al., 2019).



Consumption	CO_2e (lbs)
Air travel, 1 passenger, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000

Table 1. Estimated CO_2 consumption of common NLP models, compared to familiar consumption

The latest embodiment of this trend is the GPT-3 model developed by OpenAI, which was estimated to generate roughly the amount of carbon emissions that 5 cars would release in their entire lifespans. If this trend continues on the exponential trajectory that it's going in, it won't be long until it becomes a major antagonist in the fight against climate change.

While streamlining network architecture and GPU hardware as well as the geographic location of data centers has already been proven to yield massive energy gains (Patterson et al., 2021). This effect can only be compounded and amplified by combining it with efforts to reduce dataset complexity. In fact, many researchers have produced quite promising results while experimenting with reducing dataset size by removing un-representative and noisy data (Blachnik & Kordos, 2020; Rutkowski, 2004; Ryżko et al., 2016). Therefore trying to innovate on Instance Selection techniques is an endeavour worth pursuing in tackling the issue of inefficient energy consumption and has the potential to be a considerable step forward towards sustainability in Data Mining.

3. Background Review

In this section, I will firstly be covering a brief overview of why IS techniques were initially developed. After that, a concise taxonomy will be presented containing the categorization of several of the most important algorithms in instance selection. Finally a detailed description of 2 of the most influential IS algorithms will be given.

A brief history

Instance selection methods were originally developed with overcoming the weaknesses of the *k* nearest neighbours classifier in mind (Hart, 1968). This is why the term prototype selection is also used interchangeably with instance selection in the literature. The weaknesses associated with kNN that these prototype selection algorithms were trying to solve are the following (Igor, 2008):

1. The requirement of high storage capacity needed to store the set of training samples which allows the kNN algorithm to perform the decision rule
2. Lack of computational efficiency as every instance in the data point had to be compared with multiple other points while carrying out the decision rule in order to obtain similarities between the test and training samples.
3. Low tolerance to noise, particularly with 1NN, as it utilizes all data instances in its decision rule.

Taxonomy

Over the past few decades, several instance selection methods have emerged with different characteristics and properties. Essentially, these methods are categorized under 3 main headings: the direction of the search, the type of selection, and the evaluation of the search (Garcia et al., 2012). Understanding these properties can give us a better flavour and understanding of the type of techniques being developed in the field. The taxonomy with respect to the different properties is summarized below:

- **Direction of search**

  IS algorithms can be considered to have a search element to it and we can use it's direction to categorize certain IS algorithms.

  – incremental, starts with an empty set **P** and iteratively adds new instances to it if they fulfill a predefined criterion. Examples of this are: *CNN* and *IB2.*

  – decremental, opposite of incremental where you start with the original dataset **P** = **T** and then remove samples iteratively from **P** *HMN-E* and *HMN-EI*, *MSS*, *Drop1-5*, *RENN*.

  – batch, the input dataset is first analysed completely in "batch mode" and instances are marked for deletion. At the end of the analysis, marked instances are deleted altogether. Examples of such methods are *ENN*, *All-kNN, RNGE*, *ICF*, *CCIS*.

  - **mixed**, is a combination of incremental and decremental methods, where instances are added or dropped based on a predefined criterion. Examples: *BSE, GGA, EDA*

– **fixed**, can be seen as a subset of the mixed methods. The number of prototypes in the reduced subset **P** is given as a hyperparameter to the algorithm. This group includes the LVQ (*LVQ1*, *LVQ2.1*, *GLVQ*) family, *k-Means* and random sampling.

- **Strategy for selection**

A key part of the classification process is identifying decision boundaries. Essentially instances can be grouped into 2 types: border or central points depending on where they lie with respect to the decision boundaries (D. R. Wilson & Martinez, 1997)**.** Those that fall on the decision boundaries are called *border points* while those away from it are considered *central points.*

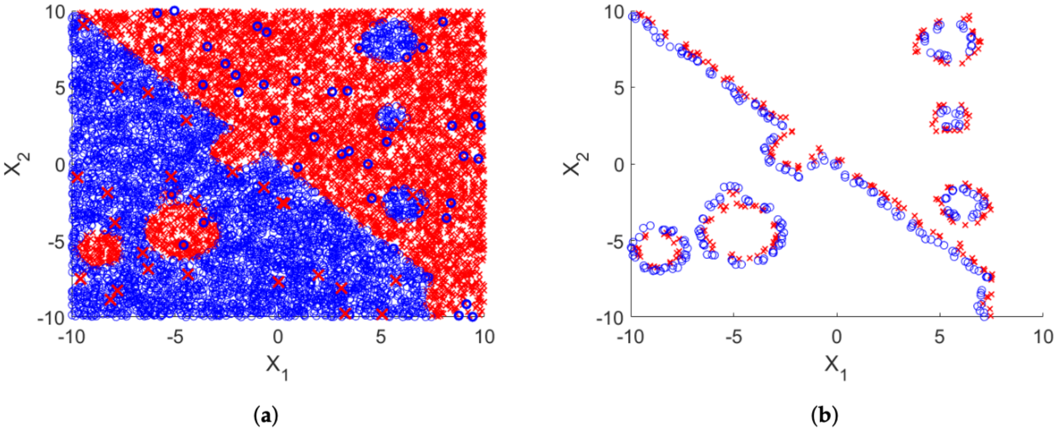


Figure 2. Effect of applying data compression (DROP3) on an example dataset **(a)** Original training set with artificial noise added **(b)** Training set after compression.

– **condensation**, this type of algorithm tries to retain border points while removing pointers further away from the decision boundaries. This usually results in a significant reduction of the dataset. Examples: *IB2*, *CNN*, *Drop1* and *Drop2*, *MSS*.

– **edition**, this type of algorithm works in the opposite way. It removes border instances and any other points that are not in agreement with their neighbours. Their main purpose is noise reduction and their compression rates are much lower than that of condensation algorithms. Examples: *ENN*, *RENN*, *All-kNN*, *HMN-E*.

– **hybrid**, these algorithms are a combination of both condensation and edition and lie somewhere between them. Most often, edition algorithms are used first for noise removal followed by condensation. Examples include: *Drop3*, *Drop5*, *ICF*, *HMN-EI*.

- **Evaluation method**

- **filters**, where the selection decision is made by internal heuristics independent of the final classifier.

- **wrappers**, where an external dedicated classifier (usually *k*NN) is used to either select or delete an instance.

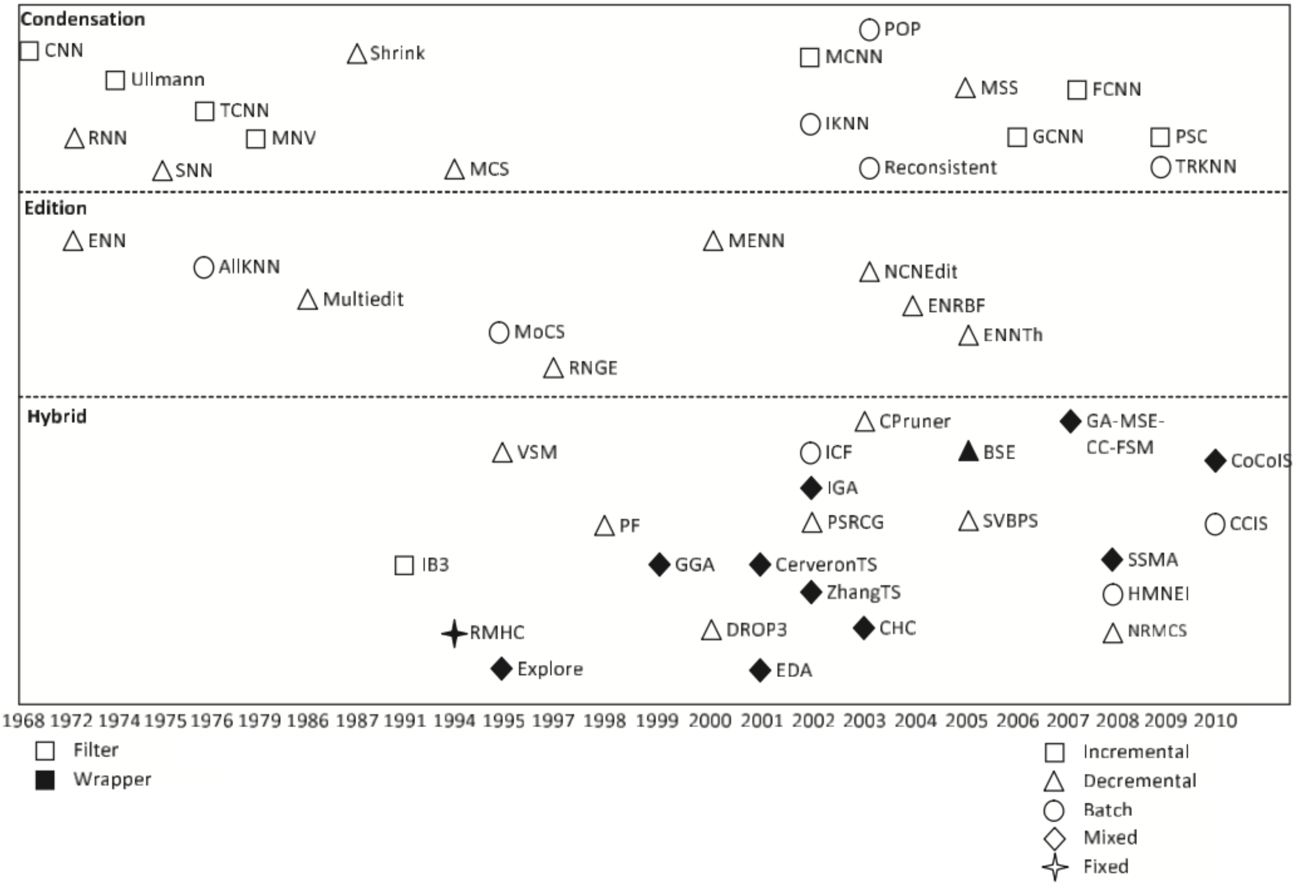


Figure 3. Instance selection methods according to the established taxonomy. Figure reproduced from (Garcia et al., 2012).

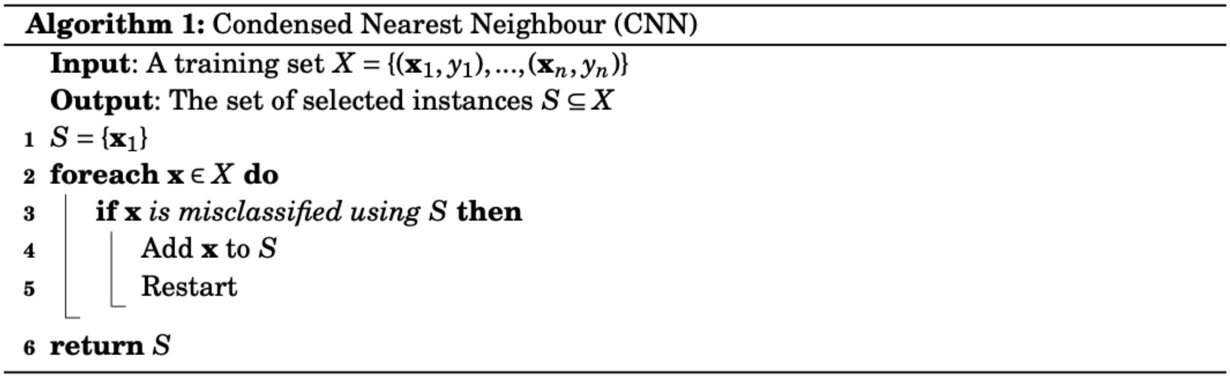
Important algorithms

According to (Garcia et al., 2012), 2 of the most influential instance selection algorithms are the following: Condensed Nearest Neighbour (CNN) and Edited Nearest Neighbour (ENN). I will briefly describe these algorithms in this section.

1. Condensed Nearest Neighbour (CNN)

Hart's Condensed Nearest Neighbours (CNN) method is widely regarded as the first explicit proposal of instance selection for the nearest neighbour rule. In this technique, consistency with regard to the training set is defined as follows: given a non-empty set X, a subset S of X is consistent with respect to X if the closest neighbour rule can properly classify all the instances in X using the subset S as a training set. If we regard the set X as the training set, a condensed subset should satisfy the characteristics of consistency and, preferably, be smaller than X, according to this definition of consistency.

The method's structure is shown in Algorithm 1. It begins with a data set, S, that contains only one randomly picked instance at first (alternatively, S could have one randomly selected instance per class in the data set). Then, using the closest neighbour criterion, it tries to categorise all of the occurrences of X by utilising the examples in S. If it succeeds, the algorithm moves on to the next instance; if it fails, the misclassified instance is added to S, and the verification of X's proper classification must start over. Upon termination, S will be returned as the chosen data set.



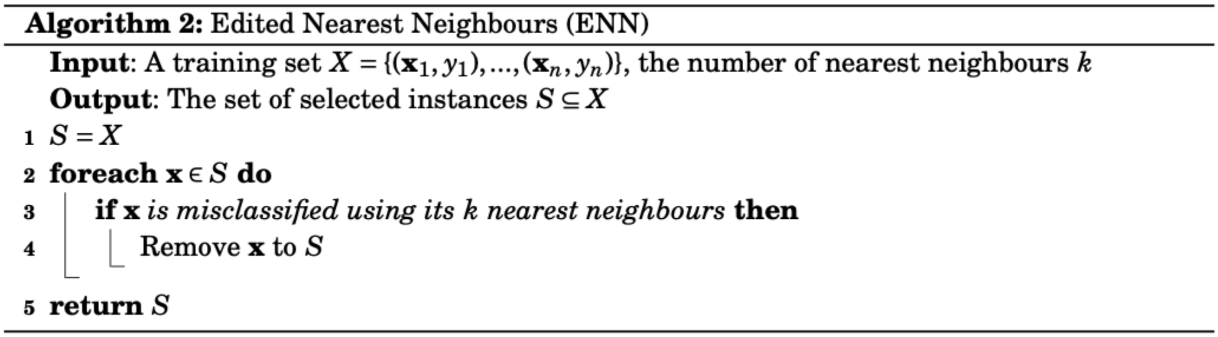
Algorithm 1: Condensed Nearest Neighbour (CNN)

```
Input: A training set $X = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$
Output: The set of selected instances $S \subseteq X$
$S = \{\mathbf{x}_1\}$
foreach $\mathbf{x} \in X$ do
    if $\mathbf{x}$ is misclassified using $S$ then
        Add $\mathbf{x}$ to $S$
        Restart
return $S$
```

2. Edited Nearest Neighbour (ENN)

(D. L. Wilson, 1972) first proposed the idea of edited datasets, which he called Edited Nearest Neighbour (ENN). It's a decremental approach, thus it starts with the entire data set X and removes each instance if its k nearest neighbours classifier is unable to classify it properly. The number of nearest neighbours, k, is an algorithm parameter. k was set to 3 in the original publication.

ENN eliminates both noisy and border cases, resulting in cleaner decision boundaries. Moreover, central instances are unaffected by the editing process. The goal of this algorithm is not to reduce the dataset, but rather to improve the accuracy of the selected subset and it is precisely due to that characteristic that it has been employed by many other algorithms for noise filtering (e.g. DROP3, ICF, etc.) The pseudocode for Algorithm 2 is shown below:



Algorithm 2: Edited Nearest Neighbours (ENN)

```
Input: A training set $X = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, the number of nearest neighbours $k$
Output: The set of selected instances $S \subseteq X$
$S = X$
foreach $\mathbf{x} \in S$ do
    if $\mathbf{x}$ is misclassified using its $k$ nearest neighbours then
        Remove $\mathbf{x}$ to $S$
return $S$
```

Experimental Design

Performance Metrics

The usefulness of IS algorithms as a general purpose training set filter is determined by a number of criteria. The compression level and prediction accuracy of the final classifier are two of the most important criteria. The compression is defined as follows:

$$compression = 1 - \frac{||P||}{||T||}$$

Where ***P*** is the subset containing the prototypes after compression and ***T*** is the original training dataset.

A greater compression value means that more samples were removed, resulting in a smaller subsequent subset P, while a lower compression value (closer to 0) indicates that the output training set is greater. The second attribute is the classifier's prediction accuracy when trained on P. This value is dependent on the applied classifier, therefore a given set P may result in high accuracy for one classifier but low accuracy for another. In this case, a basic accuracy metric was used:

$$accuracy = \frac{\#\ correctly\ classified\ samples}{\#\ total\ evaluated\ samples}$$

One additional metric that can prove to be quite useful to us is the generalization accuracy for the IS algorithms, i.e. how well they generalize to different datasets with varying sizes,

Experimental Setup

The following is a very simple initial experimental setup, it will be scaled up much more in the next iteration of this report:

Datasets selected: HTRU, Breast, Credit (all datasets were taken from the UCI repository)

Conditions*:*

- *Condition 1:* Fit *k*NN on original training data
    - Record accuracy
    - Record compression rate
    - Record Training time
- *Condition 2:* Filter using random stratified sampling (baseline algorithm)

    - Fit *k*NN on training data
    - Record accuracy
    - Record compression rate
    - Record Training time
- *Condition 3:* Filter instances using CNN (IS algo. #1)
    - Fit *k*NN on training data
    - Record accuracy
    - Record compression rate
    - Record Training time
- *Condition 4:* Filter using CRR (IS algo. #2)
    - Fit *k*NN on training data
    - Record accuracy
    - Record compression rate
    - Record Training time
- *Condition 5:* Modify CNN using k=3 (IS algo. #3)
    - Fit *k*NN on training data
    - Record accuracy
    - Record compression rate
    - Record Training time
- Calculate mean and standard deviation of results
- Calculate generalization accuracy for each algorithms

Implementation

An extremely simple small scale experiment was carried out comparing 2 IS algorithms. The 2 algorithms chosen for the experiment were the classic Condensed Nearest Neighbour (CNN) algorithm and Conservative Redundancy Removal (CRR) algorithm which is a Competence-Based Case-Base Editing approach. Essentially, case-base editing methods construct a competence model of the training data and utilize the cases' competence attributes to select which instances should be included in the edited set. (Smyth & Keane 1995) presented two crucial competence qualities for a case in a case-base: reachability and coverage sets. A case's reachability set is the set of all cases that can successfully categorise c, whereas a case's coverage set is the set of all instances that c can successfully classify. The coverage and reachability sets describe a case's local competency features and provide the foundation for a variety of editing strategies. Its pseudocode is given below:

Algorithm 4 Conservative Redundancy Removal (CRR)

```
T, Training Set
for Every c in T do                              ▷ Build case-base competence model
    CSet(c) ← Coverage Set of c
end for
/* Remove redundant cases from case-base */
ESet ← {}                                        ▷ Edited Set
TSet ← T sorted in ascending order of CSet(c) size
c ← first case in TSet
while $TSet \neq \{\}$ do
    ESet ← ESet + c
    TSet ← TSet − CSet(c)
    c ← next case in TSet
end while
```

The algorithms were used for filtering the data as part of a preprocessing step before fitting the *k*NN algorithm to it. The results were as follows:

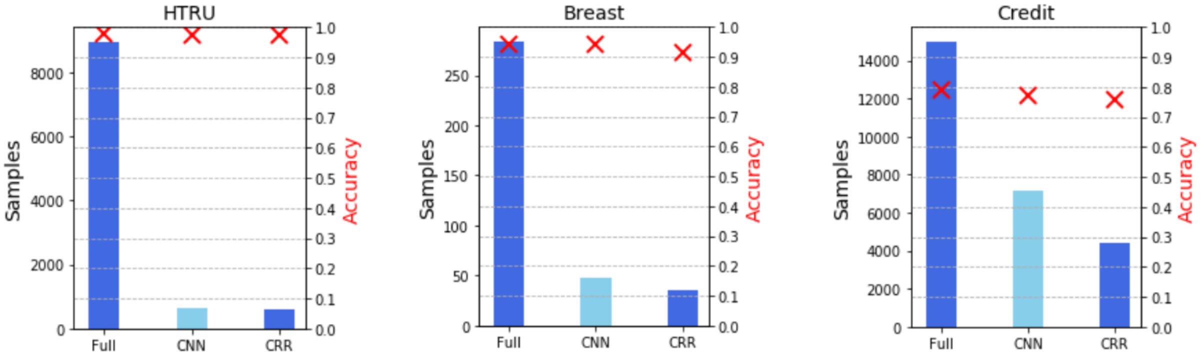
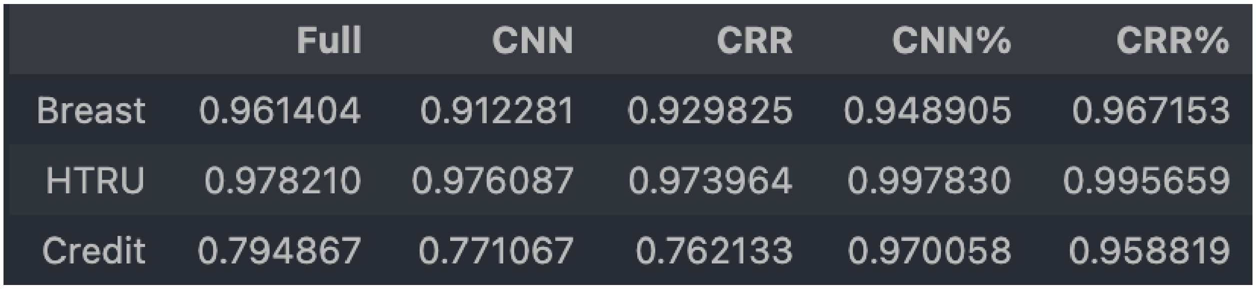
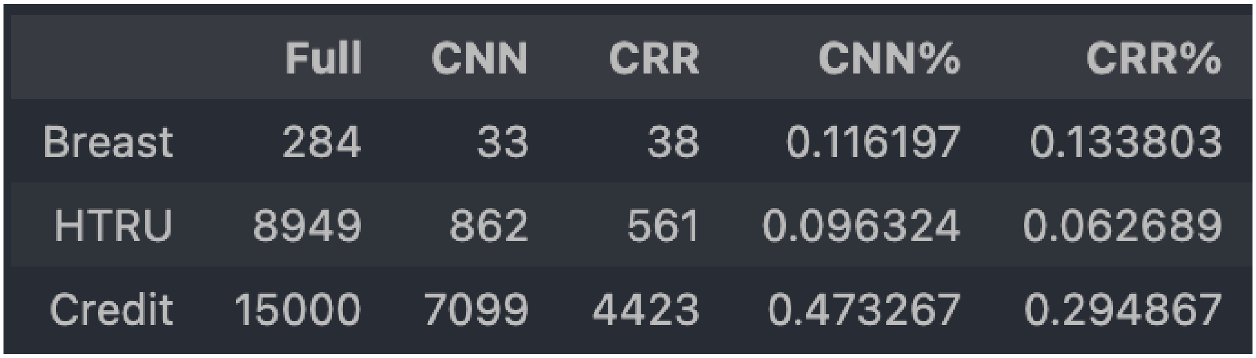


Figure 5. Comparison of set size and accuracy between CNN and CRR. (**X**) represents the accuracy.



	Full	CNN	CRR	CNN%	CRR%
Breast	0.961404	0.912281	0.929825	0.948905	0.967153
HTRU	0.978210	0.976087	0.973964	0.997830	0.995659
Credit	0.794867	0.771067	0.762133	0.970058	0.958819

Figure 6. Comparison of accuracy



	Full	CNN	CRR	CNN%	CRR%
Breast	284	33	38	0.116197	0.133803
HTRU	8949	862	561	0.096324	0.062689
Credit	15000	7099	4423	0.473267	0.294867

Figure 7. Comparison of set size

The results suggest that, for the HTRU and Breast datasets, the training set size may be shrunk considerably without affecting generalisation accuracy. When there is a lot of duplication in the training data, instance selection can provide a large speedup without sacrificing accuracy. The CRR algorithm did however perform slightly worse in terms of accuracy in the breast and credit datasets.

In order to visualize how the CNN algorithm picked instances to end up in the final dataset, I plotted a simple graph with the points selected highlighted in yellow as depicted below:

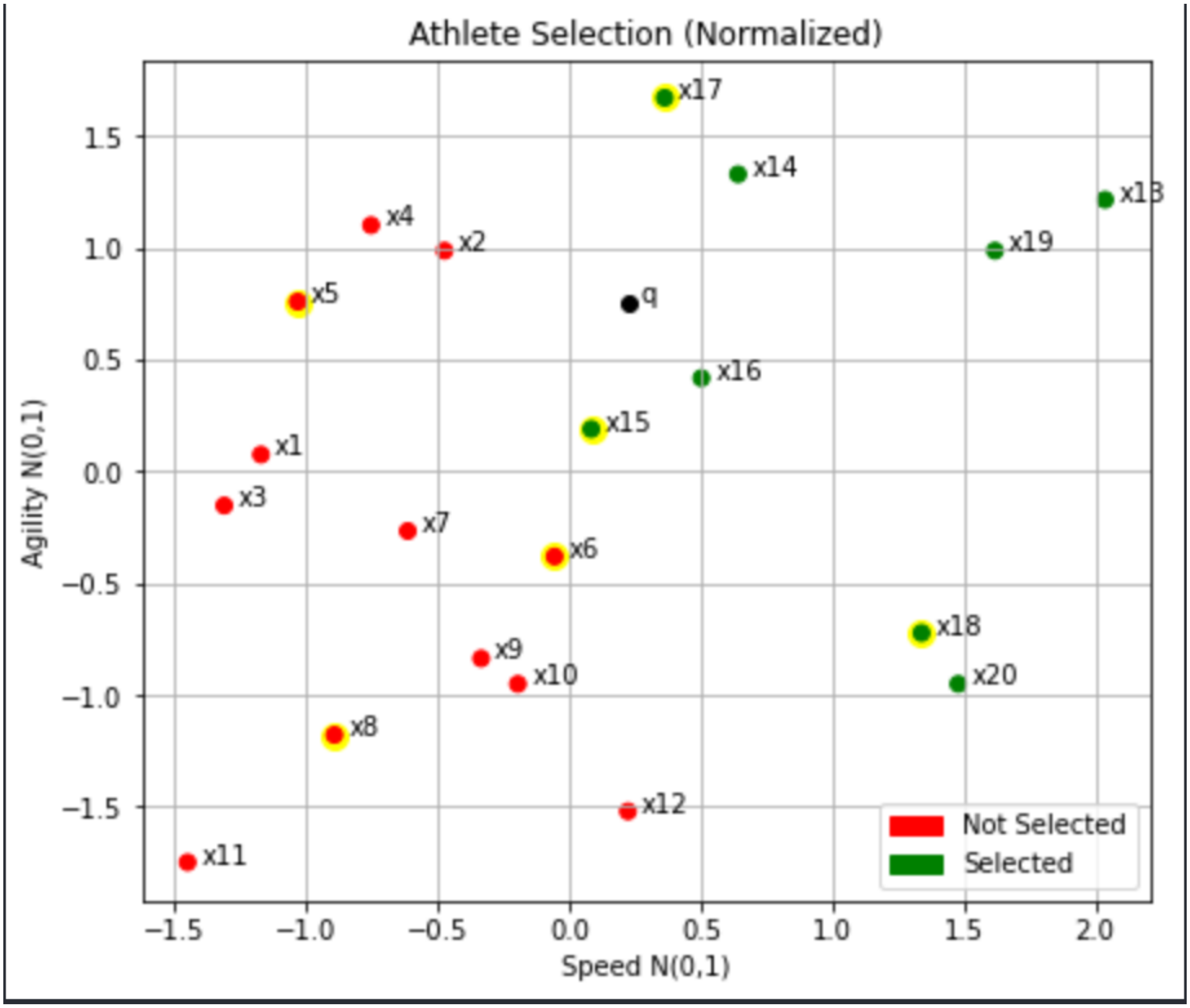


Figure 8. Instances selected by the CNN algorithm on the athlete dataset

Progress

Reflection on original work plan

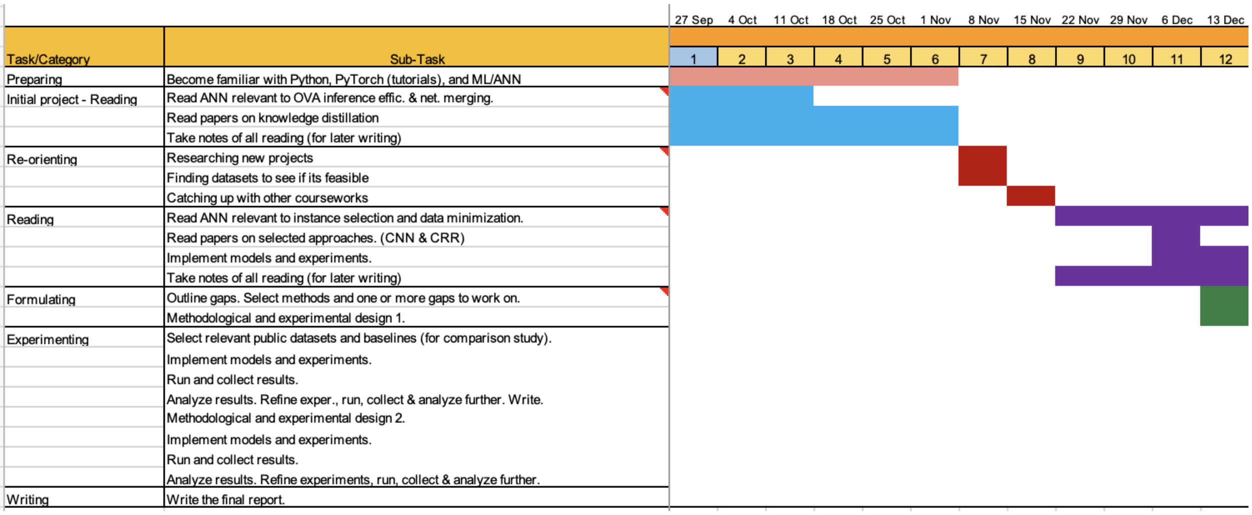
For the first 6 weeks of the semester, I was really contending with the nature of the original project that was assigned to me involving a broader scope of "Tiny Learning" involving a combination of tiny networks and tiny data. I felt I was really limited by my lack of experience in both research as well as knowledge in the subject matter. This was the first time I was exposed to a technical research in such a manner, and it seemed as though I was way out of my depth trying to navigate my way through it

I shared all these concerns and doubts that I was having with Dr. Tomas, who was very understanding and empathetic towards my predicament and very graciously guided me out of it by narrowing the scope and providing me with a very concrete project to work on with very specific objectives to try and achieve which really re-oriented me back on track and I was ready to pursue the research with renewed commitment.

Having said however, whatever progress I had made in those 6 weeks was completely reset and I had to start over from scratch. This really limited my output at the end of this semester as I only had the chance to properly work on the project for around a month, the bulk of which was spent in background reading.
Nevertheless, the negative experience in the beginning allowed me to gain a much deeper appreciation of the sea of research that existed in the realm of literature and tackling large scale projects in general.

Project Timeline

Below is the Gantt chart that I used in order to track my progress throughout the semester. Each task is split into one or more smaller subtasks so that it is more manageable to complete and it follows the SMART goal setting principles, i.e. the goal has to be specific, measurable, attainable, realistic, and time-bound.



		27 Sep	4 Oct	11 Oct	18 Oct	25 Oct	1 Nov	8 Nov	15 Nov	22 Nov	29 Nov	6 Dec	13 Dec
Task/Category	Sub-Task	1	2	3	4	5	6	7	8	9	10	11	12
Preparing	Become familiar with Python, PyTorch (tutorials), and ML/ANN	🟥	🟥	🟥	🟥	🟥	🟥						
Initial project - Reading	Read ANN relevant to OVA inference effic. & net. merging.	🟦	🟦	🟦									
	Read papers on knowledge distillation	🟦	🟦	🟦	🟦	🟦	🟦						
	Take notes of all reading (for later writing)	🟦	🟦	🟦	🟦	🟦	🟦						
Re-orienting	Researching new projects							🟥					
	Finding datasets to see if its feasible							🟥					
	Catching up with other courseworks								🟥				
Reading	Read ANN relevant to instance selection and data minimization.									🟪	🟪	🟪	🟪
	Read papers on selected approaches. (CNN & CRR)											🟪	
	Implement models and experiments.											🟪	🟪
	Take notes of all reading (for later writing)									🟪	🟪	🟪	🟪
Formulating	Outline gaps. Select methods and one or more gaps to work on.												🟩
	Methodological and experimental design 1.												🟩
Experimenting	Select relevant public datasets and baselines (for comparison study).												
	Implement models and experiments.												
	Run and collect results.												
	Analyze results. Refine exper., run, collect & analyze further. Write.												
	Methodological and experimental design 2.												
	Implement models and experiments.												
	Run and collect results.												
	Analyze results. Refine experiments, run, collect & analyze further.												
Writing	Write the final report.												

Figure 9. Gantt chart for the progress of semester 1

As you can see, a significant chunk of time was spent researching for the initial proposed plan and later re-orienting myself to the new project. There was a week in between (Week 7) where I researched several other projects to see if they could be feasible to pursue for my FYP at the suggestion of Dr Tomas. I came up with 5 shortlisted ideas:

1. Applying data science to determine how much sleep is optimal for maximising productivity
2. Image captioning
3. Detection of emotion from voice
4. Fake news detection
5. How Data Science can be applied to determine the best time for maximising crop yield

However, I quickly found out that most of them were either lacking a quality dataset to work with (and the quality of a Machine Learning model depends on the quality of the data), or they were already quite heavily researched and it would be difficult to find gaps in the literature that needed to be bridged. Therefore, at the end of that week, Dr. Tomas reassigned me the project of Tiny Learning but with a major change in scope which I was quite satisfied with as it seemed to me that the level on uncertainty with that was significantly reduced.

The week after that (Week 8) was spent catching up with the other courseworks that had piled up while I was solely focusing on finding a suitable project the week before. Finally, starting week 9 was when I got the chance to properly embark on the new project.

Future plans

As the number of credits in the following semester is going to be lower, 50 compared to 70 of this semester, I hope to fully utilize the extra time and am planning to really scale up the project, experimenting with at least 10 other IS algorithms and trying to innovate on them. While this semester focused on exploring the literature and gaining a broad understanding of the types of techniques, in the next semester, I will be going in much more depth into experimentation and specialize more into IS algorithms specific to the domain of deep learning.

In addition, future experiments that I conduct will involve multiple datasets to be used for comparison, along with using various classifiers. Overall, I hope to fully utilize the extra flexibility I will have due to the lighter workload in the next semester.

Below is a rough plan on how I choose to assign my time for the remainder of the project:

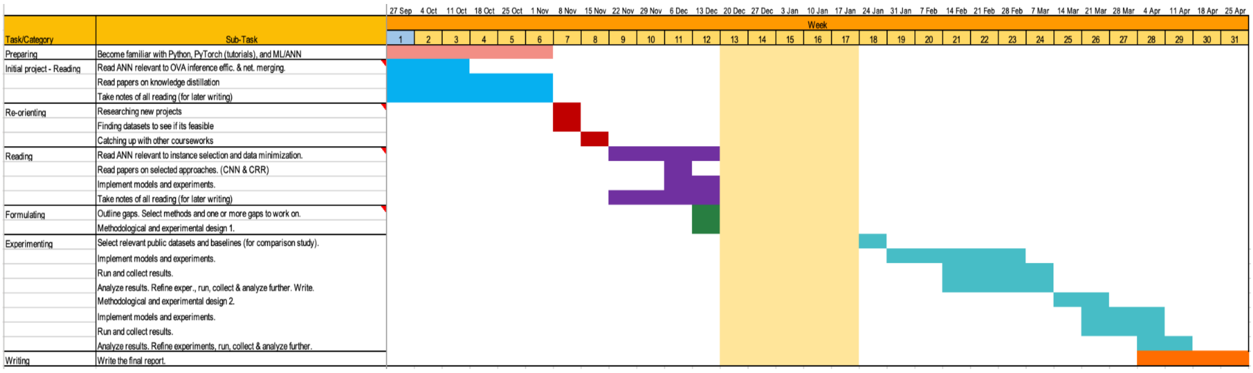


Figure 9. Gantt chart for complete duration of the project

Final Reflection

Overall, I am satisfied with the progress I made up till this stage because it was the best I could produce within the brief span of a month that I have been working on this project while balancing all my other courseworks. Furthermore, I believe that I will be able to perform much better the following semester considering I am finally clear on my objectives and aims of the project and the limited research I have done so far revealed some gaps that might be promising to explore and fill. In addition, I will have more time to explore the literature and perform experimentation and I hope that I will be able to make much more progress in the next phase.

References

Asuncion, A., Newman, D. (2007). *UCI machine learning repository*. http://www.ics.uci.edu/~mlearn/ MLRepository.html

Blachnik, M. (2015). Reducing Time Complexity of SVM Model by LVQ Data Compression. In L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, & J. M. Zurada (Eds.), *Artificial Intelligence and Soft Computing* (Vol. 9119, pp. 687–695). Springer International Publishing. https://doi.org/10.1007/978-3-319-19324-3_61

Blachnik, M. (2019). Ensembles of instance selection methods: A comparative study. *International Journal of Applied Mathematics and Computer Science*, *29*(1), 151–168. https://doi.org/10.2478/amcs-2019-0012

Blachnik, M., & Kordos, M. (2020). Comparison of Instance Selection and Construction Methods with Various Classifiers. *Applied Sciences*, *10*(11), 3933. https://doi.org/10.3390/app10113933

Chien-Hsing Chou, Bo-Han Kuo, & Fu Chang. (2006). The Generalized Condensed Nearest Neighbor Rule as A Data Reduction Method. *18th International Conference on Pattern Recognition (ICPR'06)*, 556–559. https://doi.org/10.1109/ICPR.2006.1119

Cunningham, P., & Delany, S. J. (2021). k-Nearest Neighbour Classifiers: 2nd Edition (with Python examples). *ACM Computing Surveys*, *54*(6), 1–25. https://doi.org/10.1145/3459665

Garcia, S., Derrac, J., Cano, J. R., & Herrera, F. (2012). Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 417–435. https://doi.org/10.1109/TPAMI.2011.142

García, S., Luengo, J., & Herrera, F. (2015). *Data Preprocessing in Data Mining* (Vol. 72). Springer International Publishing. https://doi.org/10.1007/978-3-319-10247-4

Hart, P. (1968). The condensed nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory*, *14*(3), 515–516. https://doi.org/10.1109/TIT.1968.1054155

Igor, K. (2008). Machine learning and data mining: Introduction to principles and algorithms. *Choice Reviews Online*, *45*(07), 45-3834-45–3834. https://doi.org/10.5860/CHOICE.45-3834

Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., & Dean, J. (2021). *Carbon Emissions and Large Neural Network Training*. 22.

Rutkowski, L. (Ed.). (2004). *Artificial intelligence and soft computing - ICAISC 2004: 7th international conference, Zakopane, Poland, June 7-11, 2004: proceedings*. Springer.

Ryżko, D., Gawrysiak, P., Kryszkiewicz, M., & Rybiński, H. (Eds.). (2016). *Machine Intelligence and Big Data in Industry* (Vol. 19). Springer International Publishing. https://doi.org/10.1007/978-3-319-30315-4

Smyth, B., & Keane, M. T. (1995). Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. *IJCAI'95: Proceedings of the 14th International Joint Conference on Artificial Intelligence*.

Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and Policy Considerations for Deep Learning in NLP. *ArXiv:1906.02243 [Cs]*. http://arxiv.org/abs/1906.02243

Wilson, D. L. (1972). Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-2*(3), 408–421. https://doi.org/10.1109/TSMC.1972.4309137

Wilson, D. R., & Martinez, T. R. (1997). Improved Heterogeneous Distance Functions. *Journal of Artificial Intelligence Research*, *6*, 1–34. https://doi.org/10.1613/jair.346