INSTITUTO TECNOLÓGICO DE AERONÁUTICA

Gabriel Barbosa Martinz

USE OF GENERATIVE NEURAL NETWORKS FOR INSTANCE SPACE CODIFICATION AND GENERATION OF DATA WITH SPECIFIC PROPERTIES

Final Paper
2023

Course of Computer Engineering

CDU 621.38

Gabriel Barbosa Martinz

USE OF GENERATIVE NEURAL NETWORKS FOR INSTANCE SPACE CODIFICATION AND GENERATION OF DATA WITH SPECIFIC PROPERTIES

Advisor

Prof^{a}. Dr^{a}. Ana Carolina Lorena (ITA)

COMPUTER ENGINEERING

São José dos Campos
Instituto Tecnológico de Aeronáutica

2023

Cataloging-in Publication Data
Documentation and Information Division

Barbosa Martinz, Gabriel
Use of generative neural networks for instance space codification and generation of data with specific properties / Gabriel Barbosa Martinz.
São José dos Campos, 2023.
30f.

Final paper (Undergraduation study) – Course of Computer Engineering– Instituto Tecnológico de Aeronáutica, 2023. Advisor: Prof^{a}. Dr^{a}. Ana Carolina Lorena.

1. Neural networks. 2. Instance space. 3. GAN. I. Instituto Tecnológico de Aeronáutica. II. Title.

BIBLIOGRAPHIC REFERENCE

BARBOSA MARTINZ, Gabriel. **Use of generative neural networks for instance space codification and generation of data with specific properties**. 2023. 30f. Final paper (Undergraduation study) – Instituto Tecnológico de Aeronáutica, São José dos Campos.

CESSION OF RIGHTS

AUTHOR'S NAME: Gabriel Barbosa Martinz
PUBLICATION TITLE: Use of generative neural networks for instance space codification and generation of data with specific properties.
PUBLICATION KIND/YEAR: Final paper (Undergraduation study) / 2023

Gabriel Barbosa Martinz
Rua H8B, 203
12.228-461 – São José dos Campos–SP

USE OF GENERATIVE NEURAL NETWORKS FOR INSTANCE SPACE CODIFICATION AND GENERATION OF DATA WITH SPECIFIC PROPERTIES

This publication was accepted like Final Work of Undergraduation Study

---

Gabriel Barbosa Martinz

Author

Ana Carolina Lorena (ITA)

Advisor

Prof. Dr. Marcos Máximo

Course Coordinator of Computer Engineering

São José dos Campos: June 23, 2023.

Aos amigos da Graduação do ITA por motivarem tanto a criação deste template pelo Fábio Fagundes Silveira quanto por motivarem a mim e outras pessoas a atualizarem e aprimorarem este excelente trabalho.

Acknowledgments

Primeiramente, gostaria de agradecer ao Dr. Donald E. Knuth, por ter desenvolvido o TeX.

Ao Dr. Leslie Lamport, por ter criado o LaTeX, facilitando muito a utilização do TeX, e assim, eu não ter que usar o Word.

Ao Prof. Dr. Meu Orientador, pela orientação e confiança depositada na realização deste trabalho.

Ao Dr. Nelson D'Ávilla, por emprestar seu nome a essa importante via de trânsito na cidade de São José dos Campos.

Ah, já estava esquecendo... agradeço também, mais uma vez ao TeX, por ele não possuir vírus de macro :-)

"If I have seen farther than others,
it is because I stood on the shoulders of giants."
— Sir Isaac Newton

Abstract

One topic of study in Machine Learning is the study of algorithmic performance and which methodologies may be used to assess this performance. A methodology known as Instance Space Analysis has been used to relate predictive performance in classification algorithms to instance hardness (how hard an instance is for an algorithm to classify). The original methodology has been defined with the instance being an entire dataset, but further work has been made to make the instance as fine-grained as an individual observation. In this work we will build upon this methodology and we propose the creation of a generative neural network model to generate new observations for a classification algorithm with predefined hardness properties.

List of Figures

List of Tables

List of Abbreviations and Acronyms

CTq	computed torque
DC	direct current
EAR	Equação Algébrica de Riccati
GDL	graus de liberdade
ISR	interrupção de serviço e rotina
LMI	linear matrices inequalities
MIMO	multiple input multiple output
PD	proporcional derivativo
PID	proporcional integrativo derivativo
PTP	point to point
UARMII	Underactuated Robot Manipulator II
VSC	variable structure control

List of Symbols

Contents

1 Introduction

1.1 Motivation

Often in a problem being tackled with Machine Learning (ML) techniques one of the most important part of the solving process is the algorithm selection. Each algorithm has a specific bias which makes it more suitable for some classes of problems than others (PAIVA *et al.*, 2022).

It is desirable, then, that we may have a way of measuring the relationship of the performance of a given algorithm in a problem with the problem's characteristics, since knowing which data is easy or difficult for a given model to classify is useful in the way that we may make changes to the original model.

(MUÑOZ *et al.*, 2018) has introduced a methodology called Instance Space Analysis (ISA), a novel way of performance evaluation and algorithm selection in classifiers by mapping the statistical properties of an instance (an entire dataset) into how difficult the instance is for the classification algorithm to perform. Further, in (PAIVA *et al.*, 2022), the methodology has been modified to have a more fine-grained analysis, with the instance being reduced to an individual observation in a classification dataset.

Given this, we can map each observation into a hardness level . One type of model that may give us new information from this data is a Generative Adversarial Network (GAN) architecture as defined by (GOODFELLOW *et al.*, 2014). This architecture is based on a zero-sum game, with a generator network trying to create data matching the original data and a discriminator network trying to discern between the original data and the generated data.

Using this, we can use the trained generator to create data with specific hardness levels and set a difficulty of classification for an entire dataset. We can use this to verify how the original model behaves with data with a given difficulty profile or to challenge the model.

1.2 Objective

This work's objective lies in providing a framework for data generation based on the relationship between instance hardness and classification performance using the GAN architecture and monitor the original model's behaviour using the generated data.

1.3 Scope

The scope of this work will be limited to exploring a GAN implementation for the generation of data, creating a Generator and a Discriminator. The modelling will be made entirely using Python, with the PyTorch (PASZKE *et al.*, 2019) framework. PyHard (PAIVA *et al.*, 2022) will be used for reproducing the ISA methodology.

1.4 Outline of this work

2 Machine Learning

This chapter will introduce Machine Learning (ML) concepts and techniques being explored in this work, namely the classification problem, neural networks and the Generative Adversarial Network architecture.

2.1 Classification

In Statistics and Machine Learning, a problem is defined as a classification problem when it consists in identifying to which categories a member of a population belongs to. An example might be identifying which race of domestic cat is shown in a picture containing a cat. An algorithm that implements classification is known as a classifier. The classifier works by analysing each observation into dependent variables and either mapping those to the categories or by comparing each observation to previous observations by means of a similarity function or loss function.

Terminology between Statistics and Machine Learning tend to differ. In this work, we will be using the terminology found in Machine Learning, namely:

- dependent variables are called features;
- categories are called classes;

In this work, we will not focus on a specific classification algorithm since ISA is not dependent on the algorithm used, only on the problem of classification.

2.2 Neural networks

Neural networks, formally called *artificial neural networks* (ANNs), are computational models inspired by networks of biological neurons (PURI *et al.*, 2016). They are made up of multiple nodes called artificial neurons that map an input to an output based on mathematical operations. This model is used extensively in ML applications because of its perceived intelligent behaviour that come from the interactions between neurons.

2.2.1 Artificial neuron

The artificial neuron is the most basic block of an ANN. It maps inputs to an output in the given fashion:

$$y = f(\mathbf{w} \cdot \mathbf{x} + b), \tag{2.1}$$

where the symbols are defined as:

$$\begin{array}{ll} \mathbf{x} = [x_1, x_2, \ldots, x_n] & \text{Input vector;} \\ \mathbf{w} = [w_1, w_2, \ldots, w_n] & \text{Weight vector;} \\ f & \text{Activation function;} \\ y & \text{Neuron output.} \end{array} \tag{2.2}$$

Figure 2.1 shows the artificial neuron model. This model of neuron is useful because it incorporates both the linear combination of input values and bias and the non-linearity of the activation function, which means it may function as a part of an universal function approximator (HORNIK *et al.*, 1989).

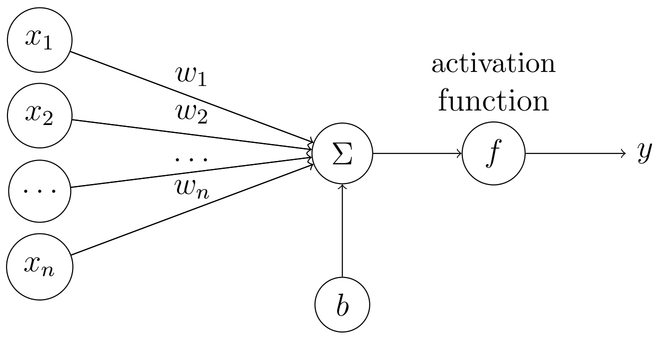

FIGURE 2.1 – Model of a neuron.

2.2.2 The network

As said before, an ANN is a network of artificial neurons. Such network may be built by having the neurons configured in layers, having each neuron in a layer connected only to neurons in either preceding or following layers or with other arrangement of connections. Figure 2.2 shows a simple model of a fully-connected (a neuron in a layer connects to every neuron in the next layer) neural network, having 3 inputs, 4 middle nodes (called a hidden layer) and 3 output nodes.

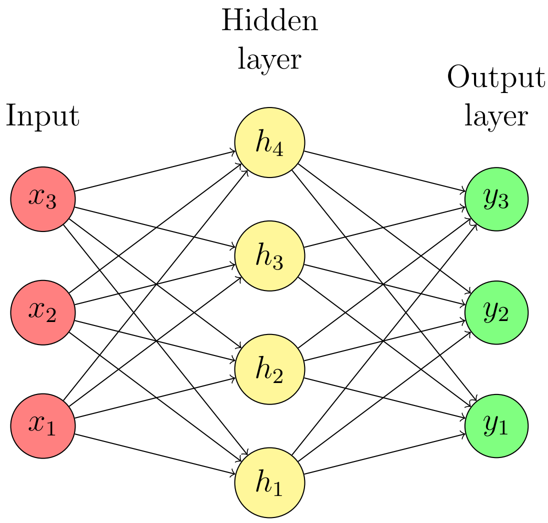

FIGURE 2.2 – A model of a simple fully-connected neural network with 3 inputs, 3 outputs and a hidden layer with 4 nodes.

2.2.3 Learning

Learning is the process by which an ANN adapts itself to a given task using data. It involves adjusting the weights of the network to improve some predefined metric (e.g. accuracy) and minimizing observed errors. In practice, learning is done by defining a loss function which is evaluated during learning and as long as its output (called loss, for short) decreases, the learning continues.

Most learning models are applications of optimization theory, like the gradient descent algorithm. In this algorithm the purpose is to find a local minima by moving against the gradient of the function. It is the basis for the Adam optimizer (KINGMA; BA, 2017), used extensively in ANNs.

2.2.3.1 Learning rate

The learning rate is a parameter in an optimization algorithm, defining the size of each step towards the local minima of a loss function. Higher learning rates shorten the learning time, but at a cost of possibly never converging and higher errors, while setting it at a value too low might have it converging in an undesirable local minimum.

2.2.3.2 Backpropagation

Backpropagation is a method to adjust the weights of the network and minimize the mean squared error. It computes the gradient of the loss function with respects to the weights and propagates backwards from the output layer to avoid redundant calculations.

2.3 Generative Adversarial Networks

A Generative Adversarial Network (GAN) is an architecture for estimating generative artificial intelligence models. It consists of a two-player minimax game between two ANNs: a generative model G and a discriminative model D. The purpose of D is to estimate the probability that a sample came from the training data instead of coming from G, and the purpose of G is to minimize that probability. A unique solution exists where D outputs the probability of $\frac{1}{2}$ everywhere (GOODFELLOW *et al.*, 2014). The generator G is, then, not trained to minimize a loss function, but to fool the discriminator D.

We will now be defining some notation for more formal modelling. Let $\mathbf{x}$ be the input data, $D(\mathbf{x})$ is then the output of the discriminator over the training data, which is the probability that the input data came from the training data rather than the generator. For the generator, let $\mathbf{z}$ be a latent space vector sampled from a standard normal distribution. $G(\mathbf{z})$ represents the generator's output, mapping $\mathbf{z}$ to the data space.

$D(G(\mathbf{z}))$ is therefore the probability that a generated input came from the training data. The goal of G is to estimate the distribution which the training data comes from (p_{data}) so that it may draw samples from this estimation (p_G) (GOODFELLOW *et al.*, 2014). The minimax loss function will be, therefore :

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{2.3}$$

In theory, the solution of this game will be when $p_G = p_{data}$ and the discriminator will guess every generated input randomly ($D(G(\mathbf{z})) = \frac{1}{2}$). Figure 2.3 shows a flowchart of this training model. Algorithm 1 is the training algorithm defined in (GOODFELLOW *et al.*, 2014).

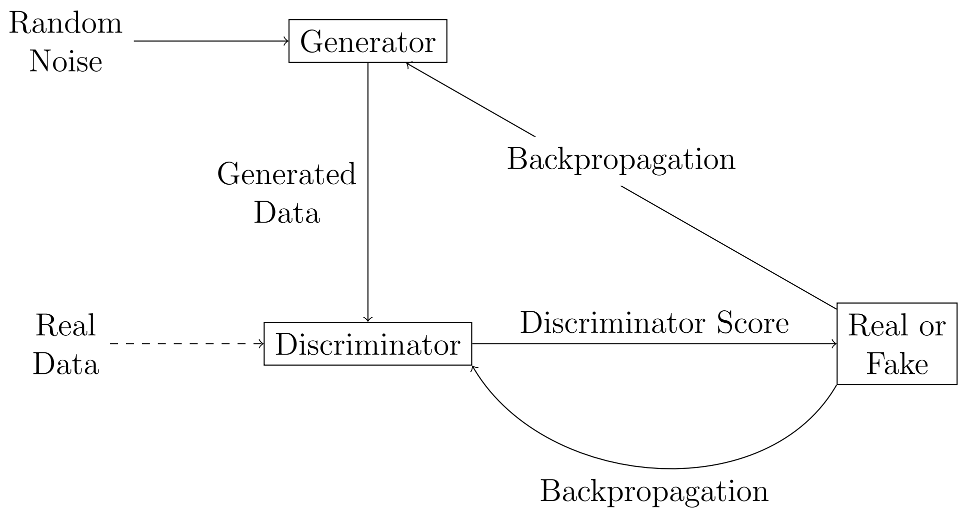

FIGURE 2.3 – Flowchart of the GAN training model.

Algorithm 1: Minibatch stochastic gradient descent training of GANs as defined in (GOODFELLOW *et al.*, 2014). The number of steps to apply to the discriminator is a hyperparameter k.

for *number of training steps* **do**
  for *k steps* **do**
    Sample minibatch of m noise samples $[\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}]$ from noise prior $p_G(\mathbf{z})$;
    Sample minibatch of m examples $[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}]$ from data distribution $p_{data}(\mathbf{x})$;
    Update the discriminator by ascending its stochastic gradient:

$$\nabla_{w_D} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\mathbf{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\mathbf{z}^{(i)}\right)\right)\right)\right];$$

  end
  Sample minibatch of m noise samples $[\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}]$ from noise prior $p_G(\mathbf{z})$;
  Update the generator by descending its stochastic gradient:

$$\nabla_{w_G} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\mathbf{z}^{(i)}\right)\right)\right);$$

end

In practice, equation 2.3 might not provide sufficient gradient for training G because of the $\log(1 - D(G(\mathbf{z})))$ term might saturate in the start of training, since D can easily differentiate between the generated data and the actual data. Instead of minimizing this term we may maximize $\log D(G(\mathbf{z}))$ to give enough gradient for G (GOODFELLOW *et al.*, 2014).

3 Instance Space Analysis

In this chapter we will be introducing the novel methodology for algorithm selection and performance evaluation called Instance Space Analysis (ISA), introduced in (MUÑOZ *et al.*, 2018). We will be showing the original definition of an instance space and the adaptation of the methodology brought up by (PAIVA *et al.*, 2022) relating instance hardness.

3.1 Instance spaces

ISA is, at its core, an extension of the Algorithm Selection Problem (ASP) (RICE, 1976). Figure 3.1 shows the ISA framework with the ASP highlighted in blue. The objective in the ASP is to automate the process of selecting algorithms based on past similar solved problems. The following sets compose the core of the ASP:

- **Problem Space** $\mathcal{P}$: contains all instances of the problem being analysed;
- **Instance Sub-space** $\mathcal{I}$: contains a subset of instances from $\mathcal{P}$ for which the characteristics and solutions are available;
- **Feature Space** $\mathcal{F}$: a set of descriptive characteristics of the instances in $\mathcal{I}$. These are also known as meta-features;
- **Algorithm Space** $\mathcal{A}$: contains algorithms that may be used to solve the instances in $\mathcal{I}$;
- **Performance Space** $\mathcal{Y}$: contains the performance evaluations of the algorithms in $\mathcal{A}$ over the instances in $\mathcal{I}$;

The combination of tuples $(x, f(x), \alpha, y(\alpha, x))$, where $x \in \mathcal{I}$, $f(x) \in \mathcal{F}$, $\alpha \in \mathcal{A}$ and $y(\alpha, x) \in \mathcal{Y}$, composes a meta-dataset $\mathcal{M}$. A meta-learner S can then be trained to select the best algorithm for an instance x based on its meta-features, that is, an algorithm α^* with maximum predictive performance for x as given by y:

$$\alpha^* = S(f(x)) = \arg\max_{\alpha} ||y(\alpha, x)||. \tag{3.1}$$

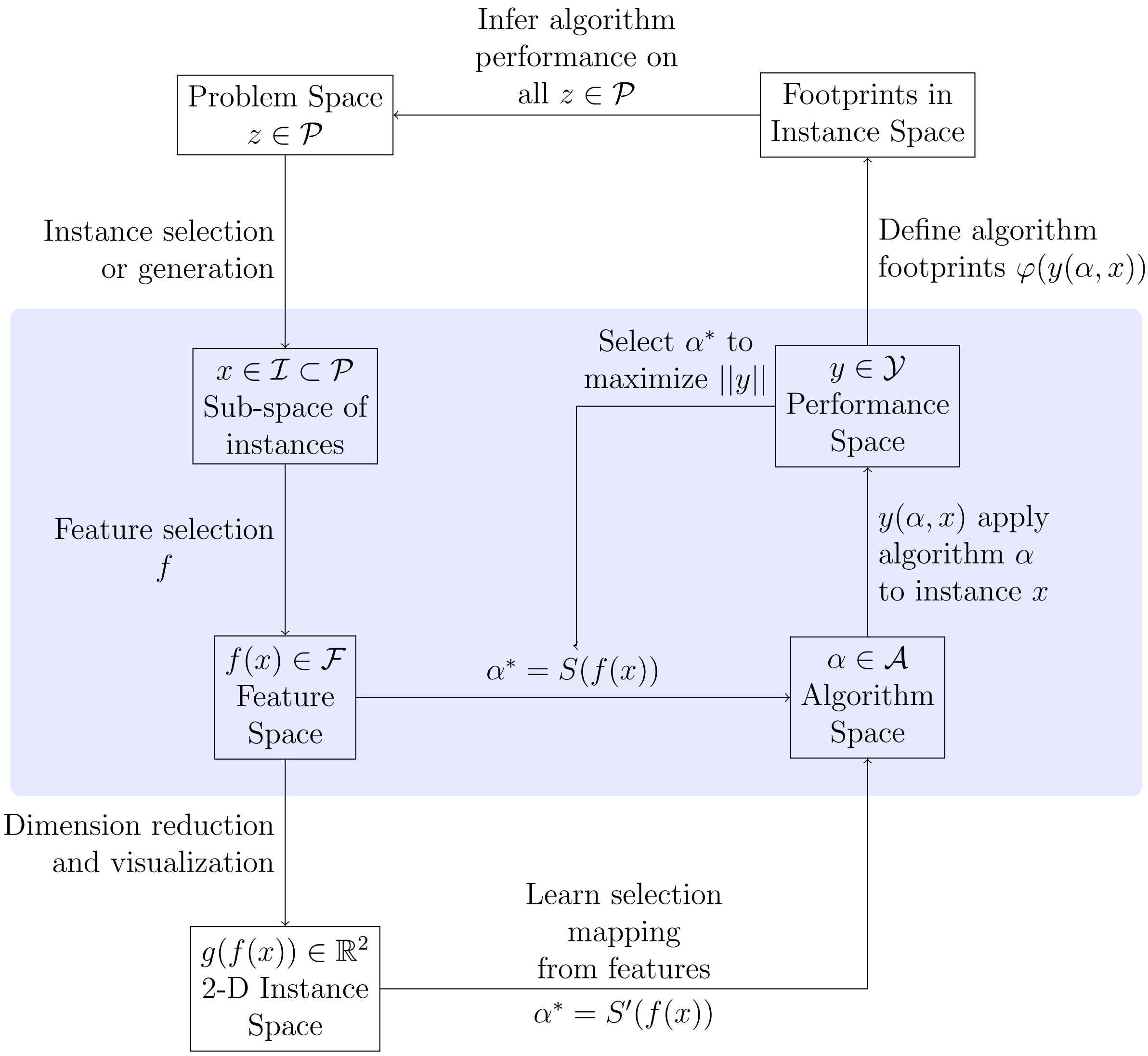

FIGURE 3.1 – ISA framework. Extracted from (MUÑOZ *et al.*, 2018). The ASP is highlighted in blue.

The ISA framework goes further to give insights into why some instances are harder to solve than others, using both the information of meta-features and algorithmic performance in a 2-D embedding, called Instance Space (IS), that can be visually inspected. An optimization problem is solved to find the mapping from meta-features to the IS $g(f(x))$, such that the distribution of algorithmic performance metrics and meta-features values display as close a linear trend as possible in the IS embedding. This embedding can then be inspected for regions of good and bad algorithmic performance and a new learner can be created to select new algorithms, as in:

$$\alpha^* = S'(g(f(x))) \tag{3.2}$$

In the IS, it is also possible to define algorithm footprints $\varphi(y(\alpha, x))$, which are areas of strength of each algorithm α. A set of objective measures can be derived from these footprints and aid in the inference of algorithmic performance for other instances that were not in $\mathcal{I}$, such as:

- the area of the footprint A, which can be normalized across multiple algorithms for comparison;
- the density of the footprint ρ, which can be calculated as the ratio between the number of instances enclosed by the footprint and its area;
- the purity of the footprint p, which is the percentage of instances in the footprint that have good performance.

Summarizing, the application of ISA requires (MUÑOZ *et al.*, 2018):

1. building the meta-dataset $\mathcal{M}$;
2. reducing the set of meta-features in $\mathcal{M}$, keeping only those able to discriminate algorithmic performance;
3. creating the 2-D IS from $\mathcal{M}$;
4. building the algorithms' footprints in the IS.

3.2 ISA for a single dataset

4 Methodology

5 Results

5.1 Planned results

6 Conclusion

6.1 Preliminary conclusions and future work

6.2 Work plan

Bibliography

GOODFELLOW, I.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A.; BENGIO, Y. Generative adversarial nets. *In*: GHAHRAMANI, Z.; WELLING, M.; CORTES, C.; LAWRENCE, N.; WEINBERGER, K. (Ed.). **Advances in Neural Information Processing Systems**. **Proceedings** [...]. Curran Associates, Inc., 2014. v. 27. Available at: https://proceedings.neurips.cc-/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural Networks**, v. 2, n. 5, p. 359–366, 1989. ISSN 0893-6080. Available at: https://www.sciencedirect.com/science/article/pii/0893608089900208.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. 2017. Available at: https://doi.org/10.48550/arXiv.1412.6980.

MUÑOZ, M. A.; VILLANOVA, L.; BAATAR, D.; SMITH-MILES, K. Instance spaces for machine learning classification. **Machine Learning**, v. 107, n. 1, p. 109–147, Jan 2018. ISSN 1573-0565. Available at: https://doi.org/10.1007/s10994-017-5629-5.

PAIVA, P. Y. A.; MORENO, C. C.; SMITH-MILES, K.; VALERIANO, M. G.; LORENA, A. C. Relating instance hardness to classification performance in a dataset: a visual approach. **Machine Learning**, v. 111, n. 8, p. 3085–3123, Aug 2022. ISSN 1573-0565. Available at: https://doi.org/10.1007/s10994-022-06205-9.

PASZKE, A.; GROSS, S.; MASSA, F.; LERER, A.; BRADBURY, J.; CHANAN, G.; KILLEEN, T.; LIN, Z.; GIMELSHEIN, N.; ANTIGA, L.; DESMAISON, A.; KöPF, A.; YANG, E.; DEVITO, Z.; RAISON, M.; TEJANI, A.; CHILAMKURTHY, S.; STEINER, B.; FANG, L.; BAI, J.; CHINTALA, S. **PyTorch: An Imperative Style, High-Performance Deep Learning Library**. 2019. Available at: https://doi.org/10.48550/arXiv.1912.01703.

PURI, M.; SOLANKI, A.; PADAWER, T.; TIPPARAJU, S. M.; MORENO, W. A.; PATHAK, Y. Introduction to artificial neural network (ann) as a predictive tool for drug design, discovery, delivery, and disposition: Basic concepts and modeling. basic concepts and modeling. **Artificial Neural Network for Drug Design, Delivery and Disposition**, Elsevier Inc., p. 3–13, 2016.

RICE, J. R. The algorithm selection problem**this work was partially supported by the national science foundation through grant gp-32940x. this chapter was presented as the george e. forsythe memorial lecture at the computer science conference, february 19, 1975, washington, d. c. *In*: RUBINOFF, M.; YOVITS, M. C. (Ed.). Elsevier, 1976, (Advances in Computers, v. 15). p. 65–118. Available at: https://www.sciencedirect.com/science/article/pii/S0065245808605203.

Appendix A - Tópicos de Dilema Linear

A.1 Uma Primeira Seção para o Apêndice

A matriz de Dilema Linear M e o vetor de torques inerciais b, utilizados na simulação são calculados segundo a formulação abaixo:

$$M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \tag{A.1}$$

FIGURE A.1 – Uma figura que está no apêndice

Annex A - Exemplo de um Primeiro Anexo

A.1 Uma Seção do Primeiro Anexo

Algum texto na primeira seção do primeiro anexo.

FOLHA DE REGISTRO DO DOCUMENTO

1. CLASSIFICAÇÃO/TIPO	2. DATA	3. DOCUMENTO Nº	4. Nº DE PÁGINAS
TC	23 de junho de 2023	DCTA/ITA/DM-018/2015	30

5. TÍTULO E SUBTÍTULO:
Use of generative neural networks for instance space codification and generation of data with specific properties

6. AUTOR(ES):
Gabriel Barbosa Martinz

7. INSTITUIÇÃO(ÕES)/ÓRGÃO(S) INTERNO(S)/DIVISÃO(ÕES):
Instituto Tecnológico de Aeronáutica – ITA

8. PALAVRAS-CHAVE SUGERIDAS PELO AUTOR:
Cupim; Cimento; Estruturas

9. PALAVRAS-CHAVE RESULTANTES DE INDEXAÇÃO:
Cupim; Dilema; Construção

10. APRESENTAÇÃO: (**X**) **Nacional** ( ) **Internacional**
Trabalho de Graduação, ITA, São José dos Campos, 2015. 30 páginas.

11. RESUMO:
Aqui começa o resumo do referido trabalho. Não tenho a menor idéia do que colocar aqui. Sendo assim, vou inventar. Lá vai: Este trabalho apresenta uma metodologia de controle de posição das juntas passivas de um manipulador subatuado de uma maneira subótima. O termo subatuado se refere ao fato de que nem todas as juntas ou graus de liberdade do sistema são equipados com atuadores, o que ocorre na prática devido a falhas ou como resultado de projeto. As juntas passivas de manipuladores desse tipo são indiretamente controladas pelo movimento das juntas ativas usando as características de acoplamento da dinâmica de manipuladores. A utilização de redundância de atuação das juntas ativas permite a minimização de alguns critérios, como consumo de energia, por exemplo. Apesar da estrutura cinemática de manipuladores subatuados ser idêntica a do totalmente atuado, em geral suas caraterísticas dinâmicas diferem devido a presença de juntas passivas. Assim, apresentamos a modelagem dinâmica de um manipulador subatuado e o conceito de índice de acoplamento. Este índice é utilizado na sequência de controle ótimo do manipulador. A hipótese de que o número de juntas ativas seja maior que o número de passivas ($n_a > n_p$) permite o controle ótimo das juntas passivas, uma vez que na etapa de controle destas há mais entradas (torques nos atuadores das juntas ativas), que elementos a controlar (posição das juntas passivas).

12. GRAU DE SIGILO:
(**X**) **OSTENSIVO** ( ) **RESERVADO** ( ) **SECRETO**