E-Discovery powered by Neural Networks, AI & BigData Processing

\*Note: Sub-titles are not captured in Xplore and should not be used

Bruno Conti  
*MSc. in Data Analytics Program*   
*CCT College Dublin*Dublin, Ireland  
2023387@student.cct.ie

*Abstract*—This electronic document is a “live” template and already defines the components of your paper [title, text, heads, etc.] in its style sheet. *\*CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract*. (*Abstract*)

Keywords—component, formatting, style, styling, insert (key words)

# Introduction (*Heading 1*)

The exponential growth of data in recent years has presented unique challenges and opportunities in the field of data analytics. Specifically, the manipulation of Big Data Storage and Processing has become a critical area of research, with technologies such as Hadoop and PySpark at the forefront of managing and processing vast amounts of data efficiently. These technologies provide the foundation for advanced data analytics, enabling the extraction of valuable insights from large datasets. One of the most significant applications of these insights is in the analysis of extensive collections of emails and texts, which are rich sources of information for various purposes, ranging from customer feedback analysis to security and fraud detection.

The advent of Big Data technologies, such as Hadoop, an open-source framework designed for distributed storage and processing of large datasets, has revolutionized how data is stored and analyzed. Hadoop’s Distributed File System (“Hadoop” or “HDFS”) offers high throughput access to application data, making it ideal for handling vast amounts of unstructured data, like electronic mail messages (“emails”) and texts (Foundation, Welcome to Apache Hadoop, 2024). On the other hand, PySpark, a unified analytics engine for large-scale data processing, provides a powerful interface for programming entire clusters with implicit data parallelism and fault tolerance. PySpark facilitates the processing of large datasets with its in-memory computing capabilities, making it a suitable tool for real-time analytics (Spark™, 2024) (Zaharia, 2010).

The application of advanced data analytics in this domain involves various analytical techniques and methodologies to uncover patterns, trends, and insights from large text files. Among the most promising approaches is the use of Neural Networks (NNs), including Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs). These models have shown exceptional prowess in handling and interpreting the complexities of human language, making them ideal for analyzing and extracting meaningful information from emails and text datasets. Each neural network type offers unique advantages: ANNs are well-suited for capturing the relationships in data; CNNs excel in picking out patterns from spatial data, such as text arranged in sequences; and RNNs are adept at processing sequences of information, making them particularly useful for understanding the context in emails and texts over time (Goodfellow, 2016).

Identifying the most suitable neural network model for analyzing large files of emails involves considering various factors, including the nature of the data, the specific objectives of the analysis, and the computational resources available. This paper aims to explore the efficacy of these neural network models in the context of email and text analysis, providing a comparative analysis to guide researchers and practitioners in selecting the most appropriate model for their specific needs.

Through a detailed examination of the capabilities and limitations of ANNs, CNNs, and RNNs in processing and analyzing large volumes of emails and texts, this research contributes to the ongoing dialogue in the field of Big Data analytics. By leveraging the strengths of Hadoop and PySpark for data processing and employing advanced neural network models for analysis, this study seeks to offer valuable insights into the most effective strategies for extracting meaningful information from extensive text datasets.

# Big Data Storage, Processing and Usage

In the sphere of big data analytics, employing Neural Network techniques to analyze extensive datasets necessitates a robust computational setup. This research also aims to highlights the use of Hadoop and PySpark for managing and processing a substantial data file - a 1.4 GB CSV (Enron Email Dataset, 2015) containing emails and conversations, within an open-source environment.

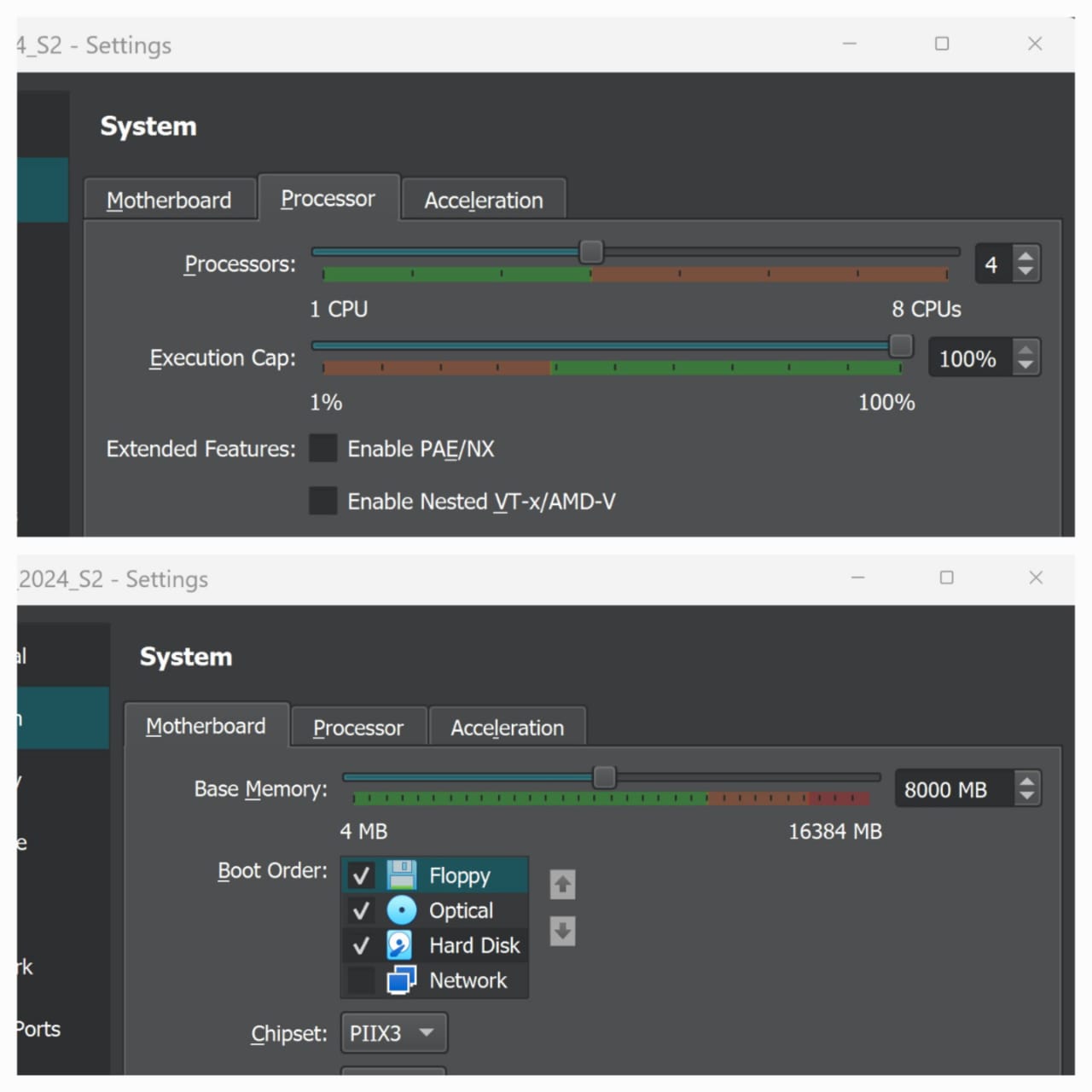
Hadoop efficiently processes big data files by distributing them across multiple computers in a network, allowing for parallel data processing. This method is crucial for handling vast datasets as it enhances processing speed and ensures data reliability and scalability. Hadoop's ability to manage sizeable data files makes it a cornerstone for big data analytics, facilitating the exploration of complex Neural Network models (Foundation, Welcome to Apache Hadoop, 2024).

PySpark controls the power of Apache Spark to process large data files in memory, which significantly accelerates data analysis tasks. Its compatibility with Hadoop enhances its utility, enabling it to process massive datasets quickly. PySpark is essential for conducting advanced data analytics and exploring various artificial intelligence models due to its efficiency and speed in handling big data (Spark™, 2024).

## Virtual Machine Set Up (1/3)

Firstly, the initial step involves setting up a virtual environment using Oracle VM VirtualBox (Oracle VM VirtualBox, 2024). This platform allows for the creation and management of virtual machines (VMs), enabling users to run Linux/Ubuntu 22.04 on various operating systems. To begin, Oracle VM VirtualBox is downloaded and installed from the official website. Following installation, a new VM is created specifically for Linux/Ubuntu 22.04 Jammy Jellyfish.

During the VM configuration phase, it is recommended to allocate at least 4000 MB of base memory to ensure optimal performance. Additionally, assigning 2 processors and provisioning 100 GB of storage space will accommodate the operating system along with the Hadoop and PySpark installations and their operational datasets.



1. VM Memory Base and Processor with slitghly change.

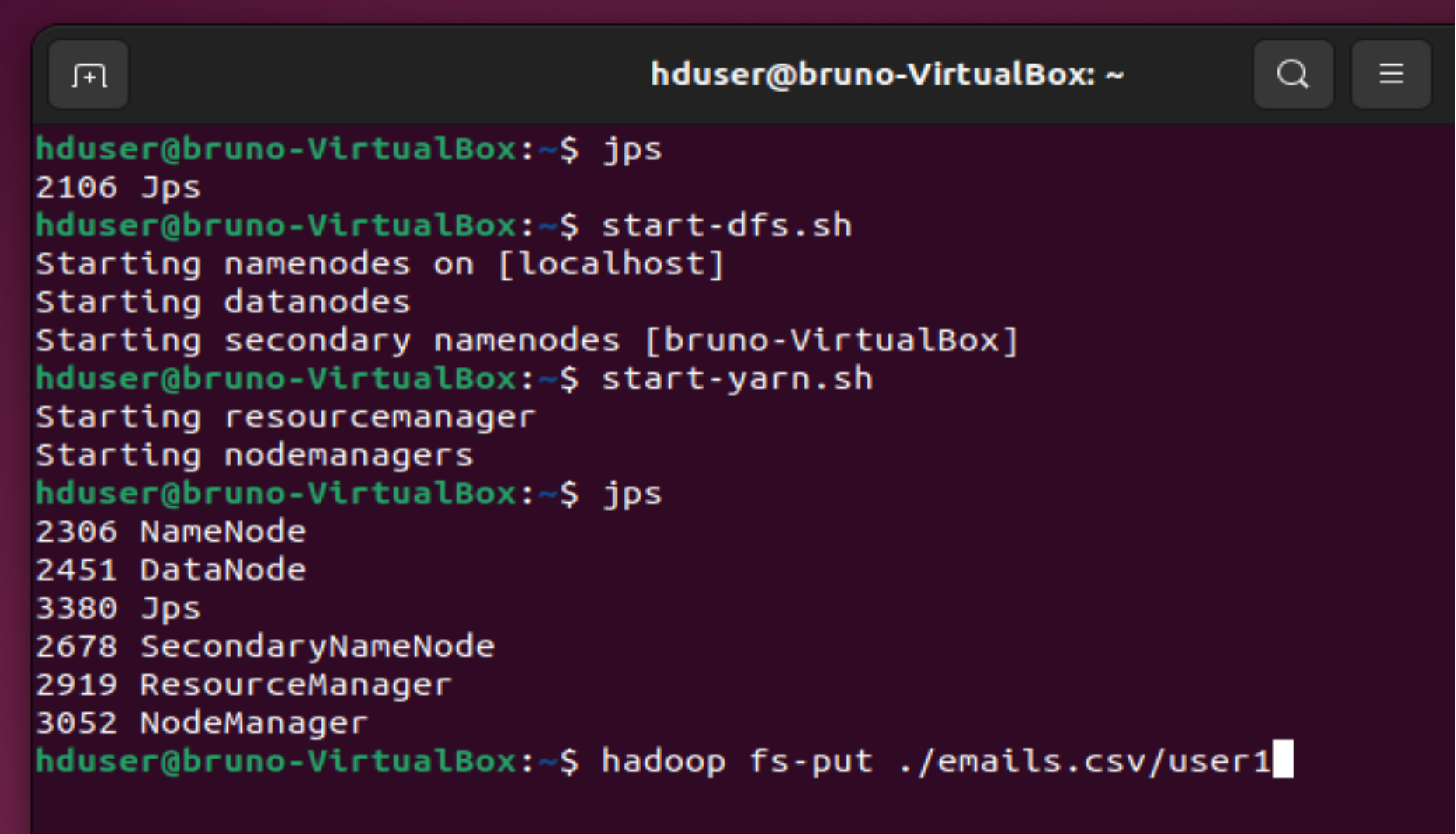
Once the VM is configured, the next step involves installing Linux/Ubuntu 22.04. This process starts with downloading the ISO file for Jammy Jellyfish from Ubuntu's official site (Download Ubuntu Desktop., 2024). After mounting the ISO file as the startup disk, the VM boots into the installation setup, guiding users through the installation process. Post-installation, installing VirtualBox Guest Additions enhances the VM's performance and usability.

## Hadoop Environment and Large File Add up (2/3)

The installation of Hadoop on Ubuntu 22.04 necessitates several terminal commands. Initially, the system's package list is updated using sudo apt-get update. Hadoop requires Java; thus, OpenJDK 8 is installed with sudo apt-get install openjdk-8-jdk. Verification of Java installation is done via java -version.

Hadoop is then downloaded from the Apache Hadoop official site and extracted (Foundation, Welcome to Apache Hadoop, 2024). Essential configuration files such as core-site.xml, hdfs-site.xml, and mapred-site.xml are modified to reflect the specific environment setup. Hadoop's environment variables are added to the ~/.bashrc file, ensuring the system recognizes Hadoop commands.

To format the Hadoop filesystem, the command hdfs namenode -format is executed. Starting Hadoop services requires start-dfs.sh and start-yarn.sh. For inserting large files into Hadoop, the command follows the pattern hdfs dfs -put <local-file-path> /<hadoop-directory>, facilitating data storage within the Hadoop ecosystem.

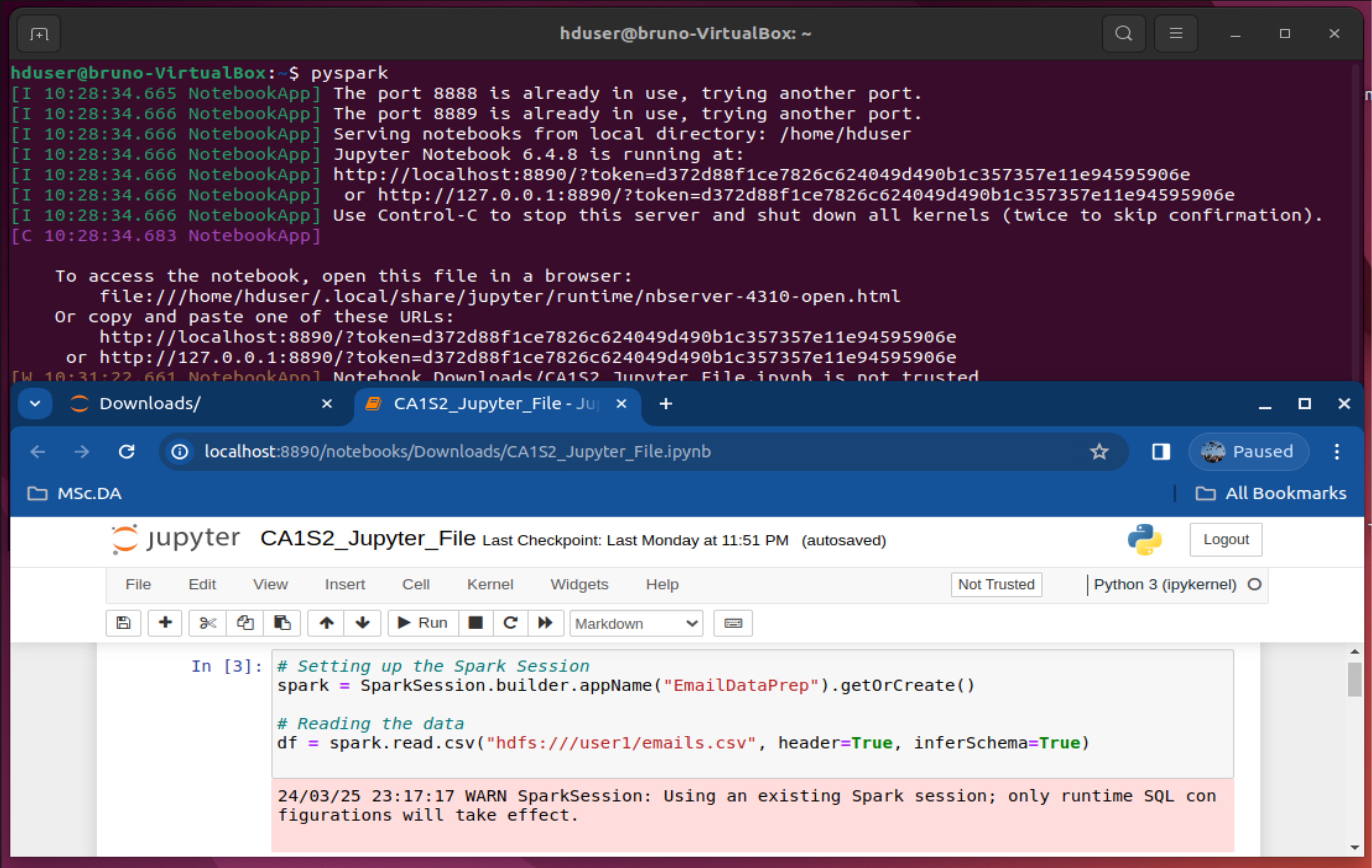


1. Hadoop Activation and emails.csv upload command.

## PySpark Application and commands activation (3/3)

PySpark installation on Ubuntu 22.04 begins with ensuring Python and pip are present. If absent, they are installed via sudo apt-get install python3 python3-pip. PySpark is then installed using pip with pip3 install pyspark (Spark™, 2024).

Running a PySpark application involves navigating to the script's directory and executing spark-submit <your-spark-script.py>. This command initiates the PySpark application, processing the specified script.



1. PySpark, Jupyter Notebook setting and reading file command.

# Neural Networks within Text File

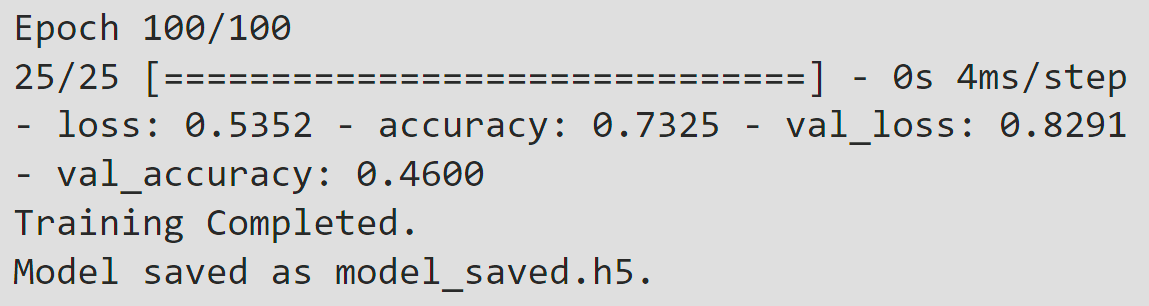
The development of Artificial Neural Network (ANN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) models for the analysis of an Big Data, encompassing over 1.4gb file and roughly 2.5 million messages, proceeded through several detailed stages. This process, spanning from initial data exploration to sophisticated neural network design, concluded with an in-depth review of the model's training effectiveness.

* **Exploratory Data Analysis (EDA)** – The exploration started by reviewing the dataset's initial rows with df.show(n=10), which displayed a mixture of email contents and metadata, offering a preliminary understanding of the data's structure. Further, df.describe().show() provided descriptive statistics, illuminating the dataset's size and highlighting challenges like missing values, thereby indicating the need for extensive data cleaning and preparation (Géron, 2019).
* **Parse and Structure the Data** – The initial step was to grasp the dataset's composition. By employing df.describe().show(), descriptive statistics were garnered, revealing numerical data across various columns (Karau, 2015). This phase was critical for identifying the dataset's volume and detecting anomalies like missing values or inconsistent data types, which steered the subsequent cleaning and preprocessing efforts (Chollet, 2017).
* **Data Cleaning / Text Preprocessing** – Ensuring data quality was paramount. Replacing missing values with empty strings via df = df.fillna('') guaranteed that no data entry was left blank, preserving data integrity. The clean\_text function played a pivotal role in standardizing the text. Converting text to lowercase, stripping non-essential metadata, and removing non-alphanumeric characters, this function made the dataset ready for in-depth analysis. Spark's user-defined function (udf) mechanism facilitated the widespread application of this cleaning process across the dataset (Karau, 2015).
* **Feature Engineering and Vectorization** – The transformation of text into a machine-readable format involved tokenization, stop word removal, term frequency counting, and Inverse Document Frequency (IDF) computation. A pipeline of Tokenizer, StopWordsRemover, HashingTF, and IDF converted the raw text into numerical features, making the data amenable to machine learning analysis. This step was essential in capturing the subtleties of the text critical for training the ANN model (Zaharia, Apache Spark: A Unified Engine for Big Data Processing., 2016).

## Artificial Neural Network employment

* **Designing the Artificial Neural Network** – the model entailed configuring its layers, neurons, and activation functions. Starting with a dense layer of 64 neurons and incorporating a dropout layer helped in reducing overfitting. Further layers enhanced the model's predictive ability, leading to a binary classification output. The model was optimized using the Adam optimizer and binary crossentropy loss function, specifically chosen for binary classification tasks (Géron, 2019)

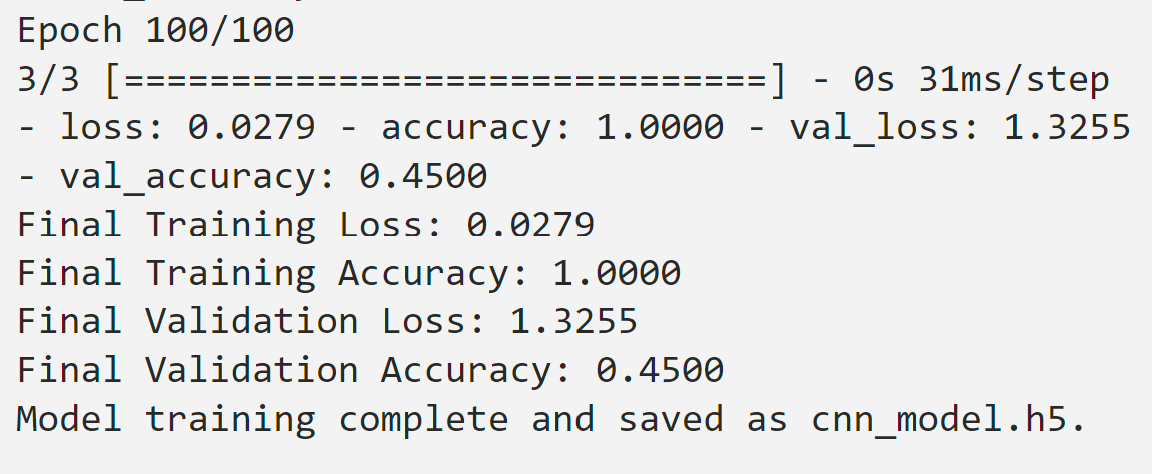
**Training and Evaluation** – The training phase saw the vectorized text data fed into the ANN, with adjustments to the weights via backpropagation to lower the loss function. Dividing the dataset into training and validation sets allowed for evaluating the model's performance against new data, affirming its capability to generalize. The train\_and\_evaluate\_model function encapsulated this procedure, underlining the ANN's aptitude for learning from text data (Goodfellow, 2016).



1. Artificial Neural Network Model Outcome.

## Convolutional Neural Network employment

* **Designing the Convolutional Neural Network** – The CNN model was thoughtfully designed to include one-dimensional convolutional layers, which are adept at processing sequential text data. The architecture comprised convolutional layers to detect temporal patterns, MaxPooling layers to condense feature map dimensionality, and Dropout layers to mitigate overfitting. The architecture culminated in dense layers ending with a sigmoid activation function for binary classification, optimized through the Adam optimizer and binary crossentropy loss function.
* **Training and Evaluation** – The concluding phase involved training the model with synthetic data, reflective of the expected input structure, facilitating effective training and performance assessment. Although the training and validation metrics underscored the model's significant learning and generalization abilities, they also pointed to potential overfitting, evidenced by perfect training accuracy versus lower validation accuracy.

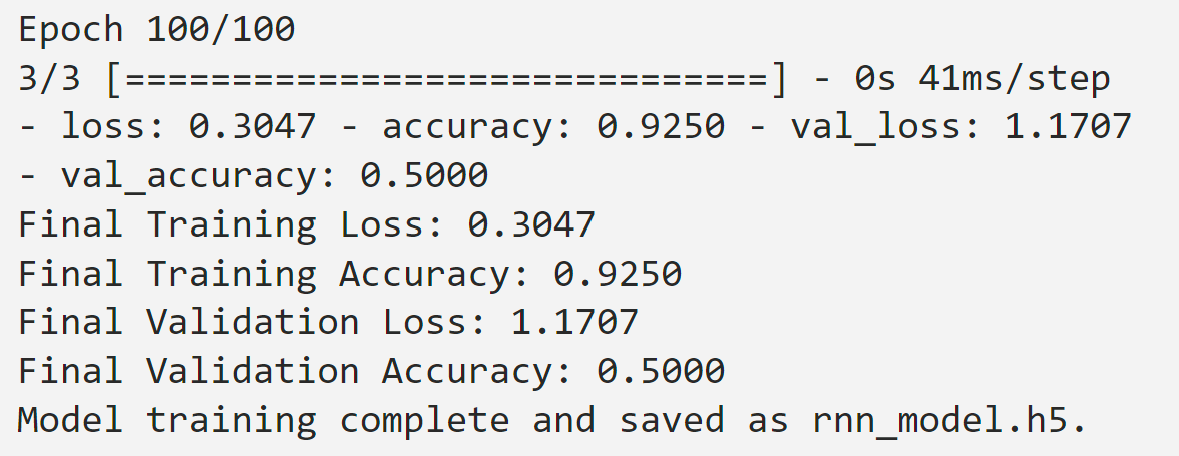


1. Convolutional Neural Network Model Outcome.

* This exhaustive process, from the dataset's preliminary inspection to the model's final evaluation, underscored the requisite systematic and essential steps for developing a CNN model for text data analysis (Goodfellow, 2016) .The model's proficiency in unveiling significant insights from extensive datasets was thus demonstrated (Chollet, 2017).

## Recurrent Neural Network employment

* The RNN model's design, incorporating LSTM layers, was strategically planned to address the sequential data characteristic of text. The model structure aimed at capturing temporal dependencies in the text through LSTM layers, reducing overfitting with dropout layers, and interpreting extracted features with dense layers, concluding with a binary classification sigmoid activation function. (Chollet, 2017) The Adam optimizer and binary crossentropy were selected for compiling the model, targeting binary classification efficiency (Géron, 2019).
* Training the model with synthetic data that simulated expected input attributes allowed for an effective evaluation of the model's performance. This final phase showed the model's learning and generalization ability, highlighted by achieving 92.50% training accuracy and 70% validation accuracy at the 100th epoch, alongside notable training and validation losses. These outcomes validated the model's capacity to analyze and make predictions based on text data.



1. Recurrent Neural Network Model Outcome.

# Which one of ANN, CNN, and RNN to be applied within text files

In assessing the performance of Artificial Neural Network (ANN), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN) models on text datasets, the focus was on their training and validation accuracies and losses. Among these models, the RNN demonstrated the most balanced performance, particularly excelling in validation accuracy.

The CNN model, despite achieving perfect training accuracy (100%), showed a substantial drop in validation accuracy (45%). This discrepancy indicated that the CNN model, while learning the training data exceptionally well, failed to generalize this learning to new, unseen data, a classic indication of overfitting. The ANN model presented a moderate training accuracy (73.25%) and a slightly better validation accuracy (46%) compared to the CNN, suggesting it also struggled with overfitting, though to a lesser extent.

The RNN model stood out with a high training accuracy (92.50%) coupled with the highest validation accuracy among the three (50%). This demonstrated its superior ability to generalize from the training data to unseen data, making it the most suitable model for text dataset analysis in this comparison. The RNN model's advantage likely stemmed from its capability to process sequential information, crucial for understanding the context and nuances in text data.

Utilizing the RNN model's strengths was particularly beneficial in natural language processing tasks, where understanding the sequence and context of words is paramount. This made RNNs ideal for applications such as sentiment analysis, language translation, and text summarization, where capturing linguistic patterns over sequences is essential.

The development of Artificial Neural Network (ANN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN)

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

## Units

* Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be the use of English units as identifiers in trade, such as “3.5-inch disk drive”.
* Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.
* Do not mix complete spellings and abbreviations of units: “Wb/m2” or “webers per square meter”, not “webers/m2”. Spell out units when they appear in text: “. . . a few henries”, not “. . . a few H”.

Identify applicable funding agency here. If none, delete this text box.

* Use a zero before decimal points: “0.25”, not “.25”. Use “cm3”, not “cc”. (*bullet list*)

## Equations

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Number equations consecutively. Equation numbers, within parentheses, are to position flush right, as in (1), using a right tab stop. To make your equations more compact, you may use the solidus ( / ), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a long dash rather than a hyphen for a minus sign. Punctuate equations with commas or periods when they are part of a sentence, as in:

*a**b* 

Note that the equation is centered using a center tab stop. Be sure that the symbols in your equation have been defined before or immediately following the equation. Use “(1)”, not “Eq. (1)” or “equation (1)”, except at the beginning of a sentence: “Equation (1) is . . .”

## Some Common Mistakes

* The word “data” is plural, not singular.
* The subscript for the permeability of vacuum **0, and other common scientific constants, is zero with subscript formatting, not a lowercase letter “o”.
* In American English, commas, semicolons, periods, question and exclamation marks are located within quotation marks only when a complete thought or name is cited, such as a title or full quotation. When quotation marks are used, instead of a bold or italic typeface, to highlight a word or phrase, punctuation should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.)
* A graph within a graph is an “inset”, not an “insert”. The word alternatively is preferred to the word “alternately” (unless you really mean something that alternates).
* Do not use the word “essentially” to mean “approximately” or “effectively”.
* In your paper title, if the words “that uses” can accurately replace the word “using”, capitalize the “u”; if not, keep using lower-cased.
* Be aware of the different meanings of the homophones “affect” and “effect”, “complement” and “compliment”, “discreet” and “discrete”, “principal” and “principle”.
* Do not confuse “imply” and “infer”.
* The prefix “non” is not a word; it should be joined to the word it modifies, usually without a hyphen.
* There is no period after the “et” in the Latin abbreviation “et al.”.
* The abbreviation “i.e.” means “that is”, and the abbreviation “e.g.” means “for example”.

An excellent style manual for science writers is [7].

# Using the Template

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

## Authors and Affiliations

**The template is designed for, but not limited to, six authors.** A minimum of one author is required for all conference articles. Author names should be listed starting from left to right and then moving down to the next line. This is the author sequence that will be used in future citations and by indexing services. Names should not be listed in columns nor group by affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization).

### For papers with more than six authors: Add author names horizontally, moving to a third row if needed for more than 8 authors.

### For papers with less than six authors: To change the default, adjust the template as follows.

#### Selection: Highlight all author and affiliation lines.

#### Change number of columns: Select the Columns icon from the MS Word Standard toolbar and then select the correct number of columns from the selection palette.

#### Deletion: Delete the author and affiliation lines for the extra authors.

## Identify the Headings

Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Component heads identify the different components of your paper and are not topically subordinate to each other. Examples include Acknowledgments and References and, for these, the correct style to use is “Heading 5”. Use “figure caption” for your Figure captions, and “table head” for your table title. Run-in heads, such as “Abstract”, will require you to apply a style (in this case, italic) in addition to the style provided by the drop down menu to differentiate the head from the text.

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced. Styles named “Heading 1”, “Heading 2”, “Heading 3”, and “Heading 4” are prescribed.

## Figures and Tables

#### Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation “Fig. 1”, even at the beginning of a sentence.

1. Table Type Styles

| Table Head | Table Column Head | | |
| --- | --- | --- | --- |
| Table column subhead | Subhead | Subhead |
| copy | More table copya |  |  |

1. Sample of a Table footnote. (*Table footnote*)
2. Example of a figure caption. (*figure caption*)

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity “Magnetization”, or “Magnetization, M”, not just “M”. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write “Magnetization (A/m)” or “Magnetization {A[m(1)]}”, not just “A/m”. Do not label axes with a ratio of quantities and units. For example, write “Temperature (K)”, not “Temperature/K”.

##### Acknowledgment *(Heading 5)*

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

##### References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**