Paper Title\* (use style: *paper title*)

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

line 1: 1st Given Name Surname   
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 4th Given Name Surname  
line 2: *dept. name of organization*  
*(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCIDline 1: 2nd Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 5th Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*

line 1: 3rd Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 6th Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

*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*)

.

# **Big data**

Big data is considered the oil of this century for its valuable contribution on many fields by providing new knowledge for better understanding to issues under study leading to innovation in many theories and technologies (Ling Tang, 2022,). Datasets contain huge volumes and complex amount of information to the extent that make it impossible to process with the traditional tools can be named as Big Data (Ishwarappa, 2015). It comes as structured, unstructured or semi structured with several different types and sources.

For this reason, big data has five characteristics: large volumes, increasing speed at which data is generated and moves around i.e. Velocity, variety of structured and unstructured data. The three V’s makes the data dirty i.e. Veracity and the last characteristic is the value of the data. To be able to implement the data, it has to be turned into value first. Therefore, value is the most important aspect of big data (Ishwarappa, 2015). After the boom in the internet Big Data can be collected from many sources on the web, Media or could (Anon., n.d.). Data formats vary from text to audio and video. The tremendous advantages of utilizing BD in decision making to increase productivity, efficient marketing, better profit in investment and forecasting research and many more remained untapped until meaningful insights are extracted from the data. The first two main processes in insight extraction are: data management to prepare the data for the analysis and data analytics to acquire knowledge form BD (Amir Gandomi, 2015). The framework of the tow processes in detail shown in **figure.1**.

The following section will discuss big data analysis and different data types and data analytical techniques for each data type.

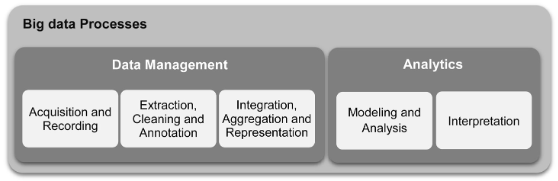
## **Big Data Analysis**

There are five phases for data analysis to obtain information, correlations and patterns form the data together with the proper tool to make informed decisions and conclusions. The phases are sequential firstly, defining the requirement gathering, to decide what and why you are doing the analysis. Then, based on the requirements you collect the data and it is either streamed or static. Third phase is the data cleaning. The fourth phase is the data analysis during which we can use data analysis tools to understand and interpret the data. The last phase is data visualization which appear in the form of charts and graph because it is easier to comprehend (Lodha, 2019,).

The next section, we review big data analytical techniques for different formats and types of data.

-**Text mining**

Text analytics, or text mining, extracts valuable information from textual data like social media posts, emails, and more. It employs statistical analysis, computation linguistics, and machine learning. Text analytics methods include information Extraction (IE) includes Entity Recognition (ER) and Relation Extraction (RE) to categorize names and identity connections between entities for informed decision-making, such as predicting.



stock market trends form financial news (Amir Gandomi, 2015). Further, includes text summarization for creating

concise document summaries, question answering in natural language, and sentiment analysis for evaluation (Feldman, 2013,) opinions toward entities. These techniques range form extractive to abstractive approaches, using advanced Natural Language Processing (NLP) methods and can be applied across various domains.

-**Speech analytics**

Audio analytics analyzes unstructured audio data primarily in customer call centers and healthcare. In call centers, it enhances customer experiences, agent performance evaluation, sales turnover, and policy compliance monitoring. Healthcare applications include diagnosing conditions affecting communication patterns and analyzing infant cries. Two common technological approaches in speech analytics are transcript-based (LVCSR) and phonetic-based systems, both involving indexing and searching phases for extracting information form audio data. LVCSR transcribes speech content into words, while phonetic-based systems work with phonemes to distinguish words by sound (Amir Gandomi, 2015).

-**Video analytics**

Also known as video content analysis (VCA), is a field that monitors, analyzes, and extract meaningful information form video streams (Amir Gandomi, 2015). It’s increasingly important due to the proliferation of CCTV cameras and video-sharing websites. Big data technologies are being used to handle the vast amount of video data, making automatic analysis possible. The primary application is in automated security and surveillance, where it efficiently detects breaches, identifies objects, and more (Hasan, 2011,). In retail, it provides business intelligence by gathering customer demographic data and monitoring store activity. Video analytics also plays a role in indexing and retrieving multimedia content, with tow system architectures: server-based edge-based, each with its advantages and drawbacks (Hu, 2011,).

As discussed above a variety of big data has been involved across various fields and predictive analytics using historical and current data to forecast future outcomes. Techniques can be divided into groups: those extrapolating historical patterns (e.g., moving averages) and those capturing interdependencies between variables (e.g., linear regression), categorized into regression and machine learning methods (Amir Gandomi, 2015) (Ling Tang, 2022,).

Handling big data efficiently involves various technologies and techniques for managing and processing. The following section will discuss the tools to storage and process big data.

**Tools to handle big data:**

Depending on the specific interests, research questions and goals the suitable tool to be employed to attack a big data is determined. There are many techniques and technologies fast growing to attempt extract insights and patterns. Discussing all of them in detail is beyond the scope of one paper, therefore, a brief listing to the fundamental and famous technology will be provided with the relevant research papers of further reading.

**First, the Hadoop framework**:

The Hadoop framework is an open-source, distributed computing platform designed for processing and storing large volumes of data across clusters of commodity hardware. It was developed to handle big data analytics and processing tasks. Hadoop provides a scalable, fault-tolerant, and cost-effective solution for managing and analyzing massive datasets (hadoop.apache.org/, n.d.) (White, 2015).

Key components of the Hadoop framework include:

Hadoop Distributed File System HDFS is a fundamental component of the Hadoop ecosystem, designed for storing and managing large volumes of data across distributed clusters. It employs a master-slave architecture with a single NameNode for metadata management and multiple DataNodes for data storage. HDFS is optimized for data reliability, fault tolerance, and scalability, making it a core technology for big data processing in Hadoop environments (Shvachko, 2010,). Furthermore, it provides a comprehensive ecosystem of tools and libraries for various data processing tasks, including real-time stream processing (with tools like Apache Kafka and Apache Flink), data warehousing (with tools like Apache Hive), and machine learning (with tools like Apache Mahout and Apache Spark ).

**Second, MapReduce** is a programming model and data processing framework introduced by Jeffrey Dean and Sanjay Ghemawat at Google. It simplifies large-scale data processing by breaking tasks into two main phases: the Map phase, where data is divided into key-value pairs and processed in parallel, and the Reduce phase, where results are aggregated. MapReduce is designed for fault tolerance and scalability, enabling efficient processing of vast datasets across distributed clusters. This approach has become a cornerstone of big data analytics and processing, with implementations like Apache Hadoop widely adopted in various industries for data-intensive tasks (Dean, 2008).

Third, YARN (Yet Another Resource Negotiator): YARN is the resource management and job scheduling component in Hadoop. It enables multi-tenancy and supports various data processing frameworks beyond MapReduce, making Hadoop more versatile for different workloads. (hadoop-yarn-site, n.d.).

Forth, Hadoop Common: This component provides libraries and utilities used by other Hadoop modules. It includes the Hadoop Distributed File System (HDFS) client, MapReduce libraries, and various other common utilities.

Fifth,Hadoop Ecosystem: Hadoop has a rich ecosystem of related projects and tools that extend its capabilities for various data processing needs. Examples include Apache Hive for data warehousing, Apache Pig for data scripting, Apache HBase for NoSQL data storage, Apache Spark for in-memory data processing, and Apache Kafka for real-time data streaming (hadoop.apache.org/, n.d.).

Hadoop is widely adopted in industries such as finance, healthcare, retail, and more for tasks like data warehousing, log processing, machine learning, and predictive analytics.

The current tools and techniques, like Hadoop, have faced several challenges when dealing with Big Data, primarily due to the evolving nature and increasing scale of data. The next section will address some of these challenges.

**Challenges**

Common challenges in Big Data analysis are crucial to address in order to effectively harness the potential of largescale data processing. These challenges include:

Heterogeneity and Incompleteness in Big Data refer to the presence of diverse and incomplete data types. This complexity requires careful handling of variations in data structure and representation during analysis. Effective data analysis must also address issues related to incomplete or erroneous data to ensure accurate results (Agrawal, 2011). The scale of Big Data poses significant challenges due to its rapid growth, surpassing available compute resources. This challenge is exacerbated by the shift towards multi-core processors and cloud computing, introducing new complexities in data management and processing efficiency. Scalability issues are particularly pronounced as traditional tools like Hadoop may struggle to effectively analyze data at petabyte or exabyte scales, highlighting the need for more scalable solutions in the Big Data landscape (Fay Chang, 2006) (Agrawal, 2011).

Timeliness: Analyzing large datasets in a timely manner is essential. Real-time or near-real-time analysis is required for applications like fraud detection. Creating efficient index structures and handling data acquisition rates are crucial aspects of timeliness (Agrawal, 2011).

Real-time Processing: Many applications require real-time or near-real-time data processing and analysis. Hadoop's batch processing model is not well-suited for these use cases (Devin, n.d.).

Data privacy: the escalating collection and linkage of personal data from diverse sources have intensified concerns regarding data privacy. Safeguarding sensitive data while conforming to privacy regulations such as GDPR presents substantial challenges. This issue is thoroughly addressed in the paper "A Survey of Big Data Architecture and Machine Learning Algorithms in Healthcare" by Qiao et al. (2016), reflecting the growing complexity of handling and analyzing data in a privacy-compliant manner within the evolving landscape of Big Data (Agrawal, 2011).

Complex Data Types: Modern data sources often include diverse and complex data types, such as multimedia and unstructured data. Traditional tools designed for structured data may struggle to handle this variety effectively.

Tool Complexity: Big Data tools like Hadoop can be complex to set up and manage. This complexity can pose challenges for organizations with limited expertise. (White, 2015)

Data Quality: Ensuring data quality and consistency is crucial for accurate analysis. Dirty or inconsistent data can lead to unreliable results.

Resource Management: Efficiently managing computing and storage resources in a distributed environment is essential for cost-effective Big Data processing.

Cost Management: Storing and processing large volumes of data can be expensive. Cost-effective strategies for data storage and processing are essential. Cost management challenges are discussed in the paper (Garofalakis, 2016)

Evolving Ecosystem: The Big Data ecosystem is rapidly evolving with new tools and technologies. Keeping up with these changes and integrating them into existing workflows can be challenging.

Security: Protecting data from security threats and ensuring data integrity are significant concerns in Big Data environments. Security challenges are discussed in the paper (José Luis Fernández-Alemán, 2013).

Big data requires new statistical approaches due to massive sample sizes, computational efficiency, and unique characteristics like heterogeneity, noise accumulation, spurious correlations, and incidental endogeneity. These challenges demand advanced statistical techniques to model and analyze big data effectively (Jianqing Fan, 2014).

Discussion

Addressing the challenges in Big Data analysis necessitates innovative approaches and advanced technologies to enhance data processing efficiency, timeliness, and privacy while accommodating human collaboration and evolving data sources. These challenges highlight the imperative for ongoing innovation within the field of Big Data, driving the creation of new tools and techniques capable of effectively tackling the intricacies of contemporary data processing.

Organizations employ various strategies and technologies to address Big Data challenges effectively. These include leveraging distributed computing frameworks like Hadoop and Apache Spark for workload distribution and enhanced scalability, utilizing data warehousing solutions such as Amazon Redshift and Google BigQuery for structured data storage, and implementing NoSQL databases like MongoDB and Cassandra for flexibility in handling unstructured data. In-memory processing technologies like Apache Ignite and SAP HANA store data in RAM, improving data retrieval and processing, especially for real-time analytics. Techniques like data compression and storage optimization reduce storage requirements, while advanced analytics and machine learning uncover insights. Robust data governance, streamlining data ingestion, scalable cloud infrastructure, data quality processes, security measures, collaboration tools, continuous monitoring, and hybrid architectures contribute to efficient Big Data handling, analysis, and decision-making.

In addition, Deep learning, a subset of machine learning, offers significant contributions to addressing Big Data challenges. It excels in automated feature extraction, eliminating the need for labor-intensive manual feature engineering. Deep learning frameworks, scalable and parallelizable, efficiently process vast datasets. These models accommodate diverse data types, including text, images, and time series, making them ideal for comprehensive analysis. For decision-making and optimization tasks, deep reinforcement learning is applicable, while real-time processing capabilities are crucial for applications like fraud detection. Deep learning also excels in anomaly detection, natural language processing, transfer learning for domain adaptation, and data privacy through techniques like federated learning. Its ability to adapt to changing data distributions suits dynamic Big Data environments effectively.

In the following sections, we will delve deeply into the topic of deep learning.

Discussion

While deep learning offers many advantages for handling Big Data challenges, it also comes with its own set of complexities, including the need for substantial computational resources, large labeled datasets for supervised learning, and potential model interpretability issues. Therefore, choosing the right deep learning approach and architecture depends on the specific Big Data problem at hand.

**Deep learning:**

Machine-learning technology, especially deep learning, is powering various aspects of modern society, from web searches to image and speech recognition. Deep learning automates the discovery of representations required for detection or classification from raw data. Its key advantage is the automatic learning of multiple levels of representation, enabling complex function learning. Deep learning has made significant advances in image and speech recognition, particle physics, brain circuit reconstruction, and natural language understanding. Its success is expected to continue due to minimal hand engineering, scalability with increased computation and data, and ongoing advancements in algorithms and architectures (LeCun, 2015).

Deep learning employs multilayer architectures composed of simple modules, most of which are trainable and compute non-linear input-output mappings. These modules progressively enhance the representation's selectivity and invariance through multiple non-linear layers. Backpropagation, introduced in the mid-1980s, is the key to training these architectures efficiently. It uses the chain rule to compute gradients by working backward from module output to input, allowing gradients to propagate through all layers. Feedforward neural network structures are common in deep learning, using non-linear functions like ReLU for fast learning. Initially overlooked, deep learning's capabilities to learn complex, multistage feature extractors with little prior knowledge. Specifically, there was a prevailing belief that basic gradient descent would become stuck in unfavorable local minima, which are weight configurations where making small adjustments would not decrease the average error (LeCun, 2015). The gradient-based learning technique is discussed thoroughly in this paper (Lecun, 1998). There are several other deep learning architectures. Deep learning is a broad field with various neural network architectures designed for specific tasks and data types. Here are some notable deep learning architectures:

**Specifically Restricted Boltzmann Machines (RBMs**) have historical importance and are relatively simple compared to other deep learning models. RBMs consist of a variant of Boltzmann machines (BMs) with stochastic visible and hidden units. They simplify the topology of the network and enhance modeling efficiency. Training RBMs typically involves the Gibbs sampler, which generates data from the RBM, and maximizing the likelihood of the RBM by adjusting weights. Gradient-based contrastive divergence (CD) algorithms are commonly used for efficient RBM training (Weibo Liu, 2017,).

**Autoencoders (AEs)** are a type of artificial neural network (ANN) used for unsupervised learning, primarily for dimensionality reduction and feature learning. AEs have been extensively researched over the past few decades, with various applications in data compression and generative modeling. They are designed to efficiently code datasets by transforming them into abstract representations and then reconstructing the original data from these representations, approximating the identity function to filter out irrelevant information and extract useful features. AEs have a one-hidden-layer feed-forward neural network structure, distinct from multilayer perceptrons (MLPs), focusing on input data reconstruction. Training involves minimizing the average reconstruction error between input and reconstructed data, with two stages: unsupervised learning for feature learning and supervised fine-tuning (Weibo Liu, 2017,).

**A standard neural network** **(NN)** comprises numerous interconnected processing units known as neurons. These neurons generate a series of real-valued activations. Input neurons become active through sensors that perceive the surrounding environment, while other neurons activate based on weighted connections from previously active neurons. Some neurons have the potential to influence the environment by initiating actions. The essence of learning or credit assignment lies in determining the appropriate weights that enable the NN to demonstrate the desired behavior, such as driving a car. Achieving this objective often involves establishing extended causal chains of computational stages, where each stage transforms the overall activation of the network, frequently in a nonlinear fashion. Deep Learning is fundamentally concerned with accurately attributing credit across numerous such stages

(Schmidhuber, 2015).

**The Convolutional Neural Network (CNN)** architecture consists of various components, including convolution layers, pooling layers, and fully connected layers. Typically, a CNN comprises a sequence of convolution layers followed by a pooling layer, and it may end with one or more fully connected layers. This transformation from input data to output through these layers is known as forward propagation. While the description here focuses on 2D-CNNs, similar operations can be adapted for three-dimensional (3D) CNNs. The convolution layer plays a central role in CNNs as it performs feature extraction through a combination of linear and nonlinear operations. This includes the convolution operation and an activation function. The convolution operation involves sliding a kernel over the input data to create a feature map. This process can be repeated with multiple kernels, generating different feature maps that represent various characteristics of the input data. These kernels effectively act as feature extractors. Key hyperparameters for the convolution operation are the kernel size and the number of kernels, which determines the depth of the output feature maps. One key feature of convolution operations is weight sharing, where kernels are shared across all image positions. This leads to several benefits, including the ability to detect local feature patterns invariant to translation, learning spatial hierarchies through down sampling and pooling, and increased model efficiency by reducing the number of parameters compared to fully connected neural networks (Yamashita, 2018) (Alzubaidi, 2021).

**Recurrent Neural Networks (RNNs**) are designed for sequential data, including time series analysis. The key concept behind RNNs is their ability to not only consider the current input but also incorporate information from previous time steps when making predictions. This approach allows RNNs to model sequences effectively, making them suitable for time series data. However, simple RNNs have limitations, such as difficulty in training and a tendency to forget important information. To address these issues, more sophisticated RNN models have been developed, with two notable ones being Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). LSTM and GRU are two popular RNN variants, each offering advantages in different scenarios. These models enable neural networks to capture and utilize long-term dependencies, which are essential for tasks involving sequential data. (Petneházi, 2019,)

. In this section will focus on the various subfields and concepts that deep learning focuses on such as:

**Deep Learning (DL) techniques** can be categorized into four main approaches: supervised learning, semi-supervised learning, unsupervised learning, and deep reinforcement learning (DRL).

**Deep Supervised Learning**: This approach deals with labeled data, where the environment has inputs and corresponding outputs. The goal is to train a model to make predictions based on input data. Common techniques in deep supervised learning include Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Deep Neural Networks (DNNs). This approach is suitable when you have labeled data, but it may struggle if the training set lacks samples for certain classes (Alzubaidi، 2021).

**Deep Semi-Supervised Learning**: In this technique, learning occurs using partially labeled datasets. Generative Adversarial Networks (GANs) and DRL are sometimes employed in this category. RNNs with architectures like Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) are also used for semi-supervised learning. The advantage is that it reduces the need for labeled data, but it may lead to incorrect decisions if irrelevant input features are present (Alzubaidi، 2021).

**Deep Unsupervised Learning**: Unsupervised learning doesn't require labeled data. Instead, the agent learns significant features or internal representations from unlabeled data to discover hidden structures or relationships. Techniques include generative networks, dimensionality reduction, and clustering. Examples of models used here are Restricted Boltzmann Machines, autoencoders, and Generative Adversarial Networks (GANs). Unsupervised learning is computationally complex and may not provide precise data sorting (Alzubaidi، 2021).

**Deep Reinforcement Learning (DRL)**: DRL is a distinct approach where an agent interacts with an environment and learns from these interactions. It's different from supervised learning, which relies on provided sample data. DRL poses challenges because there is no straightforward loss function, and the agent must learn through interaction with the environment. DRL is suitable for problems with many parameters to optimize, and it is used in applications like business strategy planning and industrial robotics automation. However, DRL can be computationally intensive and sensitive to parameter choices (Alzubaidi، 2021).

In summary, these DL approaches cater to different scenarios based on the availability of labeled data, the need for human interaction, and the complexity of the problem. Supervised learning works with labeled data, semi-supervised learning handles partially labeled data, unsupervised learning doesn't require labels, and reinforcement learning focuses on interaction with the environment. The choice of approach depends on the specific problem and its requirements.

The following section is about the practical applications of Deep Learning architectures.

**Applications:**

Deep learning architectures have found numerous practical applications due to their ability to handle vast amounts of unlabeled data, making them valuable tools for big data analysis in fields like cyber security, medical informatics, and social media. High-tech companies like Google, Facebook, and Microsoft have shown keen interest in deep learning. Several applications will be reviewed such as:

Over the past few decades, machine learning algorithms, particularly in the context of automatic speech recognition (ASR) and acoustic modeling, have played a significant role. ASR essentially involves classifying word sequences from feature sequences or speech waveforms, and it faces challenges like noisy environments, multi-modal recognition, and multilingualism. Noise removal techniques, such as spectral subtraction and Wiener filtering, are often applied to pre-process speech data. Traditional machine learning methods like Support Vector Machines (SVMs), Neural Networks (NNs), and Gaussian Mixture Models (GMMs) have been used effectively in speech recognition, especially for acoustic modeling.

**Challenges:**

**Deep learning on Big data**

Prepare Your

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections A-D below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

## 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 . . .”

Figure . Data analytics framework

* 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

(Ishwarappa, 2015)