The nanobird Relevancy Engine
A Context Analyzer and Suggestion System

Ihab Hamadeh and Zach Hajjaj
# Table of Contents

1   The nanobird Relevancy Engine  
   1.1   Definitions  
   1.2   nanobird Phrase Dictionaries  
      1.2.1   Dictionary of unambiguous phrases ($D_U$)  
      1.2.2   Dictionary of ambiguous phrases ($D_A$)  
      1.2.3   Dictionary of phrase aliases ($D_L$)  
      1.2.4   Dictionary of stop phrases ($D_S$)  
      1.2.5   Dictionary of phrase fingerprints ($D_F$)  
         1.2.5.1   Phrase Correlation Algorithm  
            1.2.5.1.1   Notations  
            1.2.5.1.2   Definitions  
            1.2.5.1.3   Examples  
   1.3   nanobird Phrase Tokenizer  
      1.3.1   Definition  
      1.3.2   Algorithm  
      1.3.3   Example  
   1.4   nanobird Context Analyzer  
      1.4.1   Definition  
      1.4.2   Algorithm  
      1.4.3   Example  
2   nRE Applications  
   2.1   Twitter and Other Social Networking Sites
2.2 News 19
2.3 Movie Recommendations 20
2.4 Book Suggestions 20
2.5 Advertisement 20
2.6 Search 20
List of Figures

Figure 1. The nanobird Relevancy Engine Components Diagram 21

Figure 2. A subset of the unambiguous phrases dictionary 22

Figure 3. An example of ambiguous phrases (apple, atmosphere, Wednesday) and their mapping to unambiguous phrases 23

Figure 4. An example of phrase aliases and their mapping to unambiguous phrases 24

Figure 5. A fingerprint of the phrase “Apple Inc.” 25

Figure 6. A Wikipedia article split into five defined sections 26

Figure 7. A Wikipedia article about “Apple Inc.” 27

Figure 8. A Wikipedia article about “iPhone” 28

Figure 9. A Wikipedia article about “Personal computer” 29

Figure 10. A Wikipedia article about “Computer software” 30

Figure 11. A Wikipedia article about “Steve Jobs” 31

Figure 12. nanobird Phrase Tokenizer Algorithm 32

Figure 13. nanobird Relevancy Engine Flowchart 33

Figure 14. An example of natural language data 34

Figure 15. Partial fingerprint of the phrase “Atmosphere of Earth” 35

Figure 16. Partial fingerprint of the phrase “Climate change” 36

Figure 17. Partial fingerprint of the phrase “Ecosystem” 37

Figure 18. Partial fingerprint of the phrase “Global warming” 38

Figure 19. Partial fingerprint of the phrase “Greenhouse gas” 39

Figure 20. An example of a nanobird Context Analyzer Graph 40

Figure 21. Example showing the interest of some Twitter users after being processed by the twitter application using nRE 41
1 The nanobird Relevancy Engine

The nanobird Relevancy Engine (nRE) is a system that analyzes an inputted natural language data (text) and extracts a set of phrases that describes and/or summarizes the text. The nRE generates a graph showing the relations between phrases and deduces the context in text. The nanobird Relevancy Engine (nRE) is composed of three parts: a set of phrase dictionaries, a phrase tokenizer, and a phrase relevancy/context analyzer (Figure 1).

1.1 Definitions

Word. A single distinct meaningful element of speech or writing, used with others (or sometimes alone) to form a sentence and typically shown with a space on either side when written or printed.

Phrase. A single word or a group of words.

Alias phrases. Phrases that take many different forms and still refer to the same thing/concept.

Ambiguous phrase. A phrase that could refer to many different thing/concept.

1.2 nanobird Phrase Dictionaries

To create a context-aware natural language analyzer, different sets of phrase dictionaries are built, each acting as a cornerstone to the nanobird Relevancy Engine. We define/build five different dictionaries: a dictionary of unambiguous phrases ($D_U$), a dictionary of ambiguous phrases ($D_A$), a dictionary of phrase aliases ($D_L$), a dictionary of stop phrases ($D_S$) and a dictionary of phrase fingerprints ($D_F$).

Each dictionary is an abstract data type composed of a collection of (key,value) pairs such that each possible key appears at most once in the collection. Two different lookups can be performed:

- a lookup to check if a given key exists in the dictionary
- a lookup to find the value (if any) that is bound to a given key

It is important to note that the sets of keys $K_{D_U}, K_{D_A}, K_{D_L}$ of the dictionaries $D_U, D_A, D_L$ respectively are unique across these dictionaries, that is:

$$K_{D_U} \sqcap K_{D_A} \sqcap K_{D_L} = \emptyset$$

1.2.1 Dictionary of unambiguous phrases ($D_U$)

Generally, a relevancy system requires a dictionary of predefined phrases obtained from different sources such as encyclopedias and glossaries. The unambiguous phrases dictionary ($D_U$) consists of predefined phrases extracted from Wikipedia, a free, web-based, collaborative, multilingual encyclopedia.
Each phrase refers to a thing/person/concept that is not open to more than one interpretation. An example of such phrases is “Billy Apple”, “Global warming”, and “Apple Inc.” which refer to something/someone specific. Figure 2 shows a subset of phrases found in the unambiguous phrases dictionary.

Each key \( k_{D_U} \) in the dictionary \( D_U \) consists of a phrase such as “Global warming” and “Apple Inc.”. The value associated with each key is not essential to the current implementation.

### 1.2.2 Dictionary of ambiguous phrases (\( D_A \))

In natural languages, some phrases are open to more than one interpretation and thus may refer to different things/concepts. Learning the different meanings of a phrase is essential to the understanding of a natural language. For instance, the word “Apple” can mean any of these following concepts:

- Apple Inc.
- Apple (fruit)
- Fiona Apple
- Apple Store

The dictionary of ambiguous phrases provides a list of ambiguous phrases along with the different possible interpretations. Each key \( k_{D_A} \) in the dictionary \( D_A \) consists of a phrase such as “Apple”. The value associated with each key is a list of unambiguous phrases in \( D_U \) (Figure 3).

\[
\{k, v\}_{D_A} = \{\text{Apple, [Apple Inc., Apple (fruit), Fiona Apple, Apple Store, ...]}\}
\]

### 1.2.3 Dictionary of phrase aliases (\( D_L \))

Some phrases may be presented in different forms and still refer to the same thing/concept. These forms range from a simple few character change to a whole phrase change. A dictionary of phrase aliases \( D_L \) is a mapping of each phrase \( k_{D_L} \) to an unambiguous phrase in \( D_U \). For instance, “Global wobble” is an alias to the phrase “Global warming”. Figure 4 gives different examples of phrase aliases and their mapping to their equivalent unambiguous phrases.

\[
\{k, v\}_{D_L} = \{\text{Climate crisis, Global warming}\}
\]
\[
\{k, v\}_{D_L} = \{\text{Planetary warming, Global warming}\}
\]
\[
\{k, v\}_{D_L} = \{\text{Stats, Statistics}\}
\]
1.2.4 Dictionary of stop phrases ($D_s$)

The dictionary of stop phrases ($D_s$) consists of phrases that are filtered out prior to or during the processing of natural language data (text). It is controlled by human input and not automated. Any group of words can be selected as the stop phrases to avoid incorrect context analysis. Examples of stop phrases are “the”, “is”, “Wednesday”, “which”, and “lots of”.

Each key ($k_{D_s}$) in the dictionary $D_s$ consists of phrases such as “the”, “is”, “Wednesday”, “which”, and “lots of”. The value associated with each key is not essential to the current implementation.

1.2.5 Dictionary of phrase fingerprints ($D_f$)

A phrase fingerprint is a map representing a phrase and its distance to other relevant phrases.

Figure 5 is an example showing the fingerprint of the phrase “Apple Inc.”.

Each key $k_{D_f}$ in the dictionary of phrase fingerprints ($D_f$) consists of a pair of phrases found in unambiguous phrase dictionary ($D_U$). The value associated with each key represents the distance between the pair of phrases.

$$\{k_j, v\}_{D_f} = \{(Apple \ Inc., iPad), 0\}$$
$$\{k_j, v\}_{D_f} = \{(Apple \ Inc., Mac OS X), 1\}$$
$$\{k_k, v\}_{D_f} = \{(Apple \ Inc., Personal \ computer), 3\}$$
$$\{k_l, v\}_{D_f} = \{(Apple \ Inc., Mac OS X \ Lion), -6\}$$

To build a dictionary of phrase fingerprints ($D_f$), encyclopedia articles, namely Wikipedia articles, are processed using a proprietary algorithm that correlates phrases within an article and across articles.

1.2.5.1 Phrase Correlation Algorithm

The nanobird Phrase Correlation Algorithm is based on building relations between phrases in Wikipedia articles.

A Wikipedia article can be split into five sections (Figure 6):

- a definition (first paragraph),
- an information box,
- a description (remaining paragraphs up to the first outline),
a summary of information (the remaining paragraphs of the article), and

- categories section.

Each section contains a set of phrases some of which are relevant to the phrase the article is about. The sets of phrases are built by collecting the phrases in the hyperlink from each section. Once the phrases are collected, they are passed to the nanobird Phrase Correlation algorithm. The algorithm is based on a set of definitions described in the remaining of this section.

1.2.5.1.1 Notations

The following notations are used in describing the Phrase Correlation algorithm:

- $\omega$: the phrase (thing/person/concept) the article is about.
- $A_\omega$: the article that defines/describes a thing, person or concept.
- $\eta_{(\omega)}^A, \eta_{(\omega)}^B, \eta_{(\omega)}^C, \eta_{(\omega)}^D, \eta_{(\omega)}^E$: the set of phrases and their aliasess in the five sections of $A_\omega$ (Figure 6).
- $\eta_{(\omega)}$: the complete set of phrases for all sections of $A_\omega$ such that
  \[ \eta_{(\omega)} = \eta_{(\omega)}^A \cup \eta_{(\omega)}^B \cup \eta_{(\omega)}^C \cup \eta_{(\omega)}^D \cup \eta_{(\omega)}^E \]
- $\lambda_{i(\omega)}^A$: the $i$th hyperlinked phrase in the “definition” $\eta_{(\omega)}^A$
- $D$: the dictionary of predefined phrases which consists of the combination of the three dictionaries $D_U, D_A, D_L$
  \[ D = D_U \cup D_A \cup D_L \]
- $\Lambda_{(\omega)}^A, \Lambda_{(\omega)}^B, \Lambda_{(\omega)}^C, \Lambda_{(\omega)}^D$: the set of hyperlink phrases in $\eta_{(\omega)}^A, \eta_{(\omega)}^B, \eta_{(\omega)}^C, \eta_{(\omega)}^D$ respectively such that:
  \[ \Lambda_{(\omega)}^A = \left\{ \lambda_{i(\omega)}^A, \lambda_{2(\omega)}^A, \ldots, \lambda_{(\omega)}^A \right\} \text{ and } \Lambda_{(\omega)}^A \subseteq \eta_{(\omega)}^A \]
  \[ \Lambda_{(\omega)}^B = \left\{ \lambda_{i(\omega)}^B, \lambda_{2(\omega)}^B, \ldots, \lambda_{(\omega)}^B \right\} \text{ and } \Lambda_{(\omega)}^B \subseteq \eta_{(\omega)}^B \]
  \[ \Lambda_{(\omega)}^C = \left\{ \lambda_{i(\omega)}^C, \lambda_{2(\omega)}^C, \ldots, \lambda_{(\omega)}^C \right\} \text{ and } \Lambda_{(\omega)}^C \subseteq \eta_{(\omega)}^C \]
  \[ \Lambda_{(\omega)}^D = \left\{ \lambda_{i(\omega)}^D, \lambda_{2(\omega)}^D, \ldots, \lambda_{(\omega)}^D \right\} \text{ and } \Lambda_{(\omega)}^D \subseteq \eta_{(\omega)}^D \]
It is important to note that all hyperlinked phrases are part of the dictionary of predefined phrases.

\[ \omega \in D, \quad \Lambda^A_{(\omega)} \subseteq D, \quad \Lambda^B_{(\omega)} \subseteq D, \quad \Lambda^C_{(\omega)} \subseteq D, \quad \Lambda^D_{(\omega)} \subseteq D \]

- \( \delta_{\omega_i \rightarrow \omega_j} \) : a signed distance between two phrases \( \omega_i \) and \( \omega_j \); thus \( \delta_{\omega_i \rightarrow \omega_j} = -\delta_{\omega_j \rightarrow \omega_i} \).

We define 15 different distances between two phrases.

\[ 0 \leq |\delta_{\omega_i \rightarrow \omega_j}| \leq 14 \]

The closer the absolute distance between two phrases, the more related those two phrases are.

### 1.2.5.1.2 Definitions

**Definition 1:** The distance between two phrases \( \omega_i \) and \( \omega_j \) is defined to be zero (\( \delta_{\omega_i \rightarrow \omega_j} := 0 \)) if the following conditions are met:
- \( \omega_i \) appears in the set of phrases of the “definition” \( \eta^A_{(\omega_i)} \) of the article \( A_{\omega_i} \), and
- \( \omega_j \) is in the set of hyperlinked phrases of the “definition” \( \Lambda^A_{(\omega_i)} \) of the article \( A_{\omega_i} \).

\[ \delta_{\omega_i \rightarrow \omega_j} := 0 \iff \omega_j \in \Lambda^A_{(\omega_i)} \text{ where } \omega_i \in \eta^A \]

**Definition 2:** The distance from phrase \( \omega_i \) to \( \omega_j \) is defined to be one (\( \delta_{\omega_i \rightarrow \omega_j} := 1 \)) if the following conditions are met:
- \( \omega_i \) appears in the set of phrases of the “information box” \( \eta^B_{(\omega_i)} \) of the article \( A_{\omega_i} \), and
- \( \omega_j \) is in the set of hyperlinked phrases of the “definition” \( \Lambda^B_{(\omega_i)} \) of the article \( A_{\omega_i} \).

**Definition 3:** \( \delta_{\omega_i \rightarrow \omega_j} := 2 \) if the following conditions are met:
- \( \omega_i \) appears in the set of phrases of the “information box” \( \eta^B_{(\omega_i)} \) of the article \( A_{\omega_i} \), and
- \( \omega_j \) is in the set of hyperlinked phrases of the “information box” \( \Lambda^B_{(\omega_i)} \) of the article \( A_{\omega_i} \).

**Definition 4:** \( \delta_{\omega_i \rightarrow \omega_j} := 3 \) if the following conditions are met:
• \( \omega_j \) appears in the set of phrases of the “description” \( (\eta^{C}_{(\omega_j)}) \) of the article \( A_{\omega_j} \), and
• \( \omega_j \) is in the set of hyperlinked phrases of the “definition” \( (\Lambda^{A}_{(\omega_j)}) \) of the article \( A_{\omega_0} \)

**Definition 5:** \( \delta_{\omega_j \rightarrow \omega_0} \coloneqq 4 \) if the following conditions are met:
• \( \omega_j \) appears in the set of phrases of the “description” \( (\eta^{C}_{(\omega_j)}) \) of the article \( A_{\omega_j} \), and
• \( \omega_j \) is in the set of hyperlinked phrases of the “information box” \( (\Lambda^{R}_{(\omega_j)}) \) of the article \( A_{\omega_0} \)

**Definition 6:** \( \delta_{\omega_j \rightarrow \omega_0} \coloneqq 5 \) if the following conditions are met:
• \( \omega_j \) appears in the set of phrases of the “description” \( (\eta^{C}_{(\omega_j)}) \) of the article \( A_{\omega_j} \), and
• \( \omega_j \) is in the set of hyperlinked phrases of the “description” \( (\Lambda^{C}_{(\omega_j)}) \) of the article \( A_{\omega_0} \)

**Definition 7:** \( \delta_{\omega_j \rightarrow \omega_0} \coloneqq 6 \) if the following conditions are met:
• \( \omega_j \) appears in the set of phrases of the “summary of information” \( (\eta^{D}_{(\omega_j)}) \) of the article \( A_{\omega_j} \), and
• \( \omega_j \) is in the set of hyperlinked phrases of the “definition” \( (\Lambda^{A}_{(\omega_j)}) \) of the article \( A_{\omega_0} \)

**Definition 8:** \( \delta_{\omega_j \rightarrow \omega_0} \coloneqq 7 \) if the following conditions are met:
• \( \omega_j \) appears in the set of phrases of the “summary of information” \( (\eta^{D}_{(\omega_j)}) \) of the article \( A_{\omega_j} \), and
• \( \omega_j \) is in the set of hyperlinked phrases of the “information box” \( (\Lambda^{R}_{(\omega_j)}) \) of the article \( A_{\omega_0} \)

**Definition 9:** \( \delta_{\omega_j \rightarrow \omega_0} \coloneqq 8 \) if the following conditions are met:
• \( \omega_j \) appears in the set of phrases of the “summary of information” \( (\eta^{D}_{(\omega_j)}) \) of article \( A_{\omega_j} \), and
• \( \omega_j \) is in the set of hyperlinked phrases of the “description” \( (\Lambda^{C}_{(\omega_j)}) \) of article \( A_{\omega_0} \)

**Definition 10:** \( \delta_{\omega_j \rightarrow \omega_0} \coloneqq 9 \) if the following conditions are met:
• \( \omega_j \) does not exist in any of the sets \( \eta^{A}_{(\omega_j)}, \eta^{B}_{(\omega_j)}, \) or \( \eta^{C}_{(\omega_j)} \) of the article \( A_{\omega_j} \),
• \( \omega_j \) is in the set of hyperlinked phrases \( \Lambda^{A}_{(\omega_j)}, \Lambda^{B}_{(\omega_j)}, \) or \( \Lambda^{C}_{(\omega_j)} \) of the article \( A_{\omega_0} \), and
• \( \omega_j \) is in the set of phrases of “categories” \( (\eta^{E}_{(\omega_j)}) \) of the article \( A_{\omega_0} \)
Definition 11: \( \delta_{\omega_j \rightarrow \omega_j} := 10 \) if the following conditions are met:
- \( \omega_j \) appears in the set of phrases of the “summary of information” \((\eta^{D}_{(\omega_j)})\) of article \( A_{\omega_j} \), and
- \( \omega_j \) is in the set of hyperlinked phrases of the “summary of information” \((\Lambda^{D}_{(\omega_j)})\) of article \( A_{\omega_0} \)

Definition 12: \( \delta_{\omega_j \rightarrow \omega_j} := 11 \) if the following conditions are met:
- \( \omega_j \) does not exist in any of the sets \( \eta^{A}_{(\omega_j)} \) of the article \( A_{\omega_j} \),
- \( \omega_j \) is in the set of hyperlinked phrases \( \Lambda^{B}_{(\omega_j)} \) of the article \( A_{\omega_0} \), and

Definition 13: \( \delta_{\omega_j \rightarrow \omega_j} := 12 \) if the following conditions are met:
- \( \omega_j \) does not exist in any of the sets \( \eta^{B}_{(\omega_j)} \) of the article \( A_{\omega_j} \),
- \( \omega_j \) is in the set of hyperlinked phrases \( \Lambda^{B}_{(\omega_j)} \) of the article \( A_{\omega_0} \), and

Definition 14: \( \delta_{\omega_j \rightarrow \omega_j} := 13 \) if the following conditions are met:
- \( \omega_j \) does not exist in any of the sets \( \eta^{C}_{(\omega_j)} \) of the article \( A_{\omega_j} \),
- \( \omega_j \) is in the set of hyperlinked phrases \( \Lambda^{C}_{(\omega_j)} \) of the article \( A_{\omega_0} \), and

Definition 15: \( \delta_{\omega_j \rightarrow \omega_j} := 14 \) if the following conditions are met:
- \( \omega_j \) does not exist in any of the sets \( \eta^{D}_{(\omega_j)} \) of the article \( A_{\omega_j} \),
- \( \omega_j \) is in the set of hyperlinked phrases \( \Lambda^{D}_{(\omega_j)} \) of the article \( A_{\omega_0} \), and

We define areas 0 to 14 \((\Omega^{(\omega)}, \Omega^{0}_{(\omega)}, \Omega^{1}_{(\omega)}, \Omega^{2}_{(\omega)}, \Omega^{3}_{(\omega)}, \Omega^{4}_{(\omega)}, \Omega^{5}_{(\omega)}, \Omega^{6}_{(\omega)}, \Omega^{7}_{(\omega)}, \Omega^{8}_{(\omega)}, \Omega^{9}_{(\omega)}, \Omega^{10}_{(\omega)}, \Omega^{11}_{(\omega)}, \Omega^{12}_{(\omega)}, \Omega^{13}_{(\omega)}, \Omega^{14}_{(\omega)})\) to be the sets of phrases relevant to the phrase \( \omega \) at distances 0 to 14.

\[
\left( \Omega^{0}_{(\omega)} \cup \Omega^{1}_{(\omega)} \cup \Omega^{2}_{(\omega)} \cup \Omega^{3}_{(\omega)} \cup \Omega^{4}_{(\omega)} \cup \Omega^{5}_{(\omega)} \cup \Omega^{6}_{(\omega)} \cup \Omega^{7}_{(\omega)} \cup \Omega^{8}_{(\omega)} \cup \Omega^{9}_{(\omega)} \cup \Omega^{10}_{(\omega)} \cup \Omega^{11}_{(\omega)} \cup \Omega^{12}_{(\omega)} \cup \Omega^{13}_{(\omega)} \cup \Omega^{14}_{(\omega)} \right) \subseteq \Lambda^{A}_{(\omega)}
\]
While phrases at distances 0 to 6 are assumed to be relevant to $\omega$, there exist some phrases at distances 7 to 14 that are not. Thus, phrases in areas 7 to 14 shall go through a phase of further relevancy testing that can be summarized as follows.

For each phrase $\alpha_{(\omega)}$ in the areas (i.e. at distance) 7 to 14, if the number of phrases (in areas 0 to 13) matching in articles $\Lambda_{(\omega)}$ is above a certain threshold $T$, then, $\alpha_{(\omega)}$ is considered relevant to $\omega$.

$$\forall \alpha \in \Omega_{(\omega)}, 7 \leq n \leq 14,$$

$$\alpha$$ is relevant to $\omega$ if and only if \[ \eta_{(\omega)} \left( \bigcup_{i=0}^{13} \Omega_{(\omega)} \right) > T \]

where $T = f(i,n,m)$ and $m = \left( \bigcup_{n=7}^{14} \Omega_{(\omega)} \right)$

Once all the definitions are applied to a Wikipedia article, a phrase fingerprint is generated (Figure 5).

1.2.5.1.3 Examples

Example 1: The word “iPhone” is in the set of hyperlinked phrases of the “definition” section ($\Lambda^A_{(Apple Inc.)}$) for the article $A_{Apple Inc.}$ describing “Apple Inc.” (Figure 7). Also, the phrase “Apple Inc.” appears in the set of phrases of the “definition” section ($\eta^A_{(iPhone)}$) for the article $A_{iPhone}$ describing “iPhone” (Figure 8). We can conclude that the distance between the two phrases “Apple Inc.” and “iPhone” is zero as per definition 1.

$$\delta_{Apple Inc. \rightarrow iPhone} := 0$$

Example 2: The phrase “Personal computer” is in the set of hyperlinked phrases of the “definition” section ($\Lambda^A_{(Apple Inc.)}$) for the article $A_{Apple Inc.}$ describing “Apple Inc.” (Figure 7). Also, the alias of the
phrase “Apple Inc.” appears in the set of phrases of the “description” section \( \eta^C_{(Personal \, computer)} \) for the article \( A_{Personal \, computer} \) describing “Personal computer” (Figure 9). We can conclude that the distance from “Apple Inc.” to “Personal computer” is three as per definition 4.

\[ \delta_{Apple \, Inc. \rightarrow \, Personal \, computer} \equiv 3 \]

**Example 3**: The phrase “Computer software” is in the set of hyperlinked phrases of the “definition” section \( \Lambda^A_{(Apple \, Inc.)} \) for the article \( A_{Apple \, Inc.} \) describing “Apple Inc.” (Figure 7). Also, the alias of the phrase “Apple Inc.” appears in the set of phrases of the “description” section \( \eta^C_{(Computer \, software)} \) for the article \( A_{Computer \, software} \) describing “Computer software” (Figure 10). We can conclude that the distance from “Apple Inc.” to “Computer software” is six as per definition 7.

\[ \delta_{Apple \, Inc. \rightarrow \, Computer \, software} \equiv 6 \]

**Example 4**: The phrase “Apple Inc.” is in the set of hyperlinked phrases of the “definition” section \( \Lambda^A_{(Steve \, Jobs)} \) for the article \( A_{Steve \, Jobs} \) describing “Steve Jobs” (Figure 11). Also, the phrase “Steve Jobs” appears in the set of phrases of the “information box” section \( \eta^B_{(Apple \, Inc.)} \) for the article \( A_{Apple \, Inc.} \) describing “Apple Inc.” (Figure 7). We can conclude that the distance from “Steve Jobs” to “Apple Inc.” is one as per definition 2.

\[ \delta_{Steve \, Jobs \rightarrow Apple \, Inc.} \equiv 1 \quad \Leftrightarrow \quad \delta_{Apple \, Inc. \rightarrow \, Steve \, Jobs} \equiv -1 \]

1.3 nanobird Phrase Tokenizer

1.3.1 Definition

The nanobird Phrase Tokenizer is responsible for splitting an inputted text into sets of phrases. The tokenizer dissects an inputted text and generates three different sets of phrases \( S_U, S_A, S_L \). The phrases in those sets consist of longest exact matches found in any of the unambiguous phrase, ambiguous phrase, or phrase aliases dictionaries \( D_U, D_A, D_L \). Those sets are, at a later stage, used by the nanobird Context Analyzer (as depicted in Figure 13) to analyze the relation among phrases in those sets and study their contextual correlation.
1.3.2 Algorithm

An inputted text is first split into an ordered set of words \( S_o \) using the space character as a divider between different words.

The \textit{nanobird Phrase Tokenizer}'s algorithm starts by forming a phrase \( P \) from the first word \( \omega_i \), located at position \( i \), in the ordered set of words \( S_o \):

\[
i \leftarrow 1, \quad P \leftarrow \omega_i, \quad \omega_i \in S_o
\]

The phrase \( P \) is tested against the following four conditions and different actions are taken.

**Condition 1:** if an exact match of \( P \) is found in any of the dictionaries \( D_U, D_A, D_L \), then the phrase is temporarily saved for future use \(( P_{\text{saved}} \leftarrow P )\) and the position of the last word used is temporarily saved in \( i_{\text{saved}} \), i.e. \( i_{\text{saved}} \leftarrow i \). Moreover, a new phrase \(( P \leftarrow P \oplus \omega_{i+1} )\) is formed by concatenating the next word \( \omega_{i+1} \) with the current phrase \( P \). The new phrase is then tested again against the four conditions.

**Condition 2:** if a submatch of \( P \) is found in any of the dictionaries \( D_U, D_A, D_L \), then a new phrase \(( P \leftarrow P \oplus \omega_{i+1} )\) is formed by concatenating the next word \( \omega_{i+1} \) with the current phrase \( P \). The new phrase is then tested again against the four conditions. The position of the next word to continue processing from becomes \( i \leftarrow i + 1 \).

**Condition 3:** if neither an exact match nor a submatch of \( P \) is found in any of the dictionaries \( D_U, D_A, D_L \) and there exists a temporarily saved phrase \( P_{\text{saved}} \) (as described in condition 1) such that \( P_{\text{saved}} \neq \emptyset \), then \( P_{\text{saved}} \) is considered to be the longest matching phrase found in any of the dictionaries \( D_U, D_A, D_L \) and thus, is added to one of sets of phrases \( S_U, S_A, \) or \( S_L \) depending on whether the exact match was found in \( D_U, D_A, \) or \( D_L \) respectively.

\[
P_{\text{saved}} \in D_U \quad \Rightarrow \quad S_U \leftarrow S_U \cup \{ P_{\text{saved}} \}
\]

\[
P_{\text{saved}} \in D_A \quad \Rightarrow \quad S_A \leftarrow S_A \cup \{ P_{\text{saved}} \}
\]

\[
P_{\text{saved}} \in D_L \quad \Rightarrow \quad S_L \leftarrow S_L \cup \{ P_{\text{saved}} \}
\]

At this point, more actions are taken:
The nanobird Phrase Tokenizer firsts generates the ordered set of words $S_O$ from the above given text.
For instance, an inputted text shown in Figure 14 will generate the following ordered set of words.

\[ S_o = \{ \text{Steven, Jobs, requested, that, the, Apple, store, on, Chestnut,} \}
\{ \text{Street, in, San, Francisco, be, closed, next, Monday.} \} \]

The tokenizer looks up the dictionaries \( D_U, D_A, D_L \) and finds a submatch in \( D_L \). As per condition 2, a new phrase is formed by concatenating the next word with the current phrase. Thus,

\[ P \leftarrow \text{Steven Jobs} \]

The tokenizer looks up \( P \) in the dictionaries \( D_U, D_A, D_L \) and finds an exact match in \( D_L \). As per condition 1, then the phrase is temporarily saved for future use and the position of the last word used is temporarily saved in \( i_{\text{saved}} \). A new phrase is formed by concatenating the next word:

\[ P_{\text{saved}} \leftarrow \text{Steven Jobs} \]
\[ i_{\text{saved}} \leftarrow 2 \]

\[ P \leftarrow \text{Steven Jobs requested} \]

The tokenizer looks up \( P \) in the dictionaries \( D_U, D_A, D_L \) and finds neither an exact match nor a submatch. Thus, as per condition 3, \( P_{\text{saved}} \) is considered to be the longest matching phrase found in dictionary \( D_L \), and thus, is added to the set of phrases \( S_L \).

\[ P_{\text{saved}} \in D_L \Rightarrow S_L \leftarrow \{ \text{Steven Jobs} \} \]

\( P_{\text{saved}} \) is deleted.

\[ P_{\text{saved}} \leftarrow \emptyset \]

The position of the next word to continue processing from is learned (\( i \leftarrow i_{\text{saved}} + 1 \))

\[ i \leftarrow 3 \]

The new phrase is now:

\[ P \leftarrow \text{requested} \]
A new lookup is done on $P$ in the dictionaries $D_U, D_A, D_L$ and neither an exact match nor a submatch is found and there exists no temporarily saved phrase that is $P_{saved} = \emptyset$. As per condition 4, the position of the next word to continue processing from is learned ($i \leftarrow i_{saved} + 1$)

$$i \leftarrow 4$$

The new phrase is now:

$$P \leftarrow \text{that}$$

The processing of the text continues until the three sets $S_U, S_A, S_L$ are extracted. For the above given text, the three generated sets are:

- $S_U = \{\text{Apple store, San Francisco}\}$
- $S_A = \{\text{Chestnut, Monday}\}$
- $S_L = \{\text{Steven Jobs}\}$

### 1.4 nanobird Context Analyzer

#### 1.4.1 Definition

The nanobird Context Analyzer is responsible for generating the nanobird Context Graph, a graph depicting the relations between the different phrases extracted by the nanobird Phrase Tokenizer. The graph’s vertices consist of phrases, while its edges represent the distances between phrases. By looking at the graph and the correlation between the different phrases extracted from a given text, it becomes possible to learn the most important phrases that describe/outline/summarize that given text.

#### 1.4.2 Algorithm

The nanobird Context Analyzer takes, as an input, the sets of phrases $S_U, S_A, S_L$ created by passing an inputted natural language data (text) to the nanobird Phrase Tokenizer.

Each phrase in each set is then filtered out when compared against the sets of keys of the stop phrases dictionary $K_{D_U}$. Any phrase matching any of the keys in $K_{D_U}$ is removed from the three sets. This results in new sets of phrases $S'_U, S'_A, S'_L$. 
The resulting set of ambiguous phrases \( S_A \) is then converted into a set of unambiguous phrases \( S^U_A \) using dictionary \( D_A \).

Similarly, the set of phrase aliases \( S_L \) is then converted into a set of unambiguous phrases \( S^U_L \) using dictionary \( D_L \). Then, the three sets of unambiguous phrases are merged into a single set \( S \):

\[
S = S'_U \bigcap S^A_U \bigcap S^L_U
\]

Every pair of phrases from the set \( S \) is looked up in the fingerprint dictionary \( D_F \) and the distance between the pair of phrases is extracted. Once all the distances are extracted between every pair of phrases, a graph is generated by the nanobird Context Analyzer. The nanobird Context Analyzer internal flowchart is shown in Figure 13.

### 1.4.3 Example

The following example goes step by step through this context analysis process.

An example of a natural language data is given in Figure 14. The three sets of phrases generated by passing the text through the nanobird Phrase Tokenizer (as described in section 1.3) are:

\[
\begin{align*}
S'_U &= \{\text{Climate change}\} \\
S_A &= \{\text{atmosphere, Wednesday}\} \\
S'_L &= \{\text{climate crisis, greenhouse gases, ecosystems}\}
\end{align*}
\]

Assume the following to be a predefined stop phrases dictionary:

\[
D_S = \{\text{who, best, where, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, once}\}
\]

The three sets are first filtered using the stop phrases dictionary \( D_S \), yielding the following sets where the word “Wednesday” is omitted since it is part of stop phrases dictionary.

\[
\begin{align*}
S'_U &= \{\text{Climate change}\} \\
S'_A &= \{\text{atmosphere}\} \\
S'_L &= \{\text{climate crisis, greenhouse gases, ecosystems}\}
\end{align*}
\]

The new set \( S'_A \) is then transformed using dictionaries \( D_A \) (refer to Figure 3) into the set \( S^U_A \). For instance, the word “atmosphere” is replaced with the phrases “Atmosphere of Earth”, “Atmosphere”, etc.

Similarly, the new set $S_{L}^{'}$ is then transformed using dictionaries $D_{L}$ (refer to Figure 4) into the set $S_{L}^{U}$. For instance, the phrase "climate crisis" is replaced with the phrase "Global warming", while the phrase "ecosystems" is replaced with the phrase "Ecosystem".

The new resulting sets are thus:

$S_{U}^{'} = \{\text{Climate change}\}$

$S_{d}^{U} = \{\text{Atmosphere of Earth, Atmosphere,}, \text{ Adobe Atmosphere, Atmosphere Visual Effects}\}$

$S_{L}^{U} = \{\text{Global warming, Greenhouse gas, Ecosystem}\}$

The three sets of unambiguous phrases are merged into a single set $S$.

Using the dictionary of phrase fingerprints $D_{F}$, the phrases in the set $S$ are correlated. Figure 15 depicts a fingerprint map of the phrase “Atmosphere of Earth”. As shown in Figure 15, the phrase “Atmosphere of Earth” is related to the phrase “Climate change” positioned at a distance 10.

$$\delta_{\text{Atmosphere of Earth \rightarrow Climate change}} := 10$$

In Figure 16, a relation exists between the two words “Climate change” and “Global warming”, such that:

$$\delta_{\text{Climate change \rightarrow Global warming}} := -3 \iff \delta_{\text{Global warming \rightarrow Climate change}} := 3$$

Similarly, the maps of phrase fingerprints depicted in Figure 17, Figure 18, and Figure 19 yield the following relations:

$$\delta_{\text{Ecosystem \rightarrow Global warming}} := 10$$

$$\delta_{\text{Global warming \rightarrow Atmosphere}} := 14$$
On the other hand, no relations were found between the phrase “Adobe Atmosphere” and the other phrases of the set $S$. As a result, the phrase “Adobe Atmosphere” is discarded. Finally, a nanobird Context Graph is generated (Figure 20) showing only the phrases for which correlations with other phrases are found using the fingerprints. Those phrases describe/outline/summarize the natural language data of Figure 14.

2 nRE Applications

The application of nRE is very rich, ranging from profiling social media users to delivering tailored advertisements. The following sections summarize some of the applications where nRE could play a big part.

2.1 Twitter and Other Social Networking Sites

Pinpointing interests of social media (Facebook, Twitter, and Google+) users has been considered to be a very challenging problem. Using nRE, profiling users has been proven to be feasible. The solution opens the door to predicting what a user will want to read within their social media feeds and what news articles he/she will be interested in reading in the future.

A Twitter application using nRE has been developed where twitter accounts are processed and profiled. For each account, the last 100 tweets are collected and each tweet is passed to nRE to yield the set of phrases that best describe the tweets or the links within the tweets. Then, the occurrence of the same phrases in all the 100 sets (generated by all the tweets) is noted down and the phrases with the highest number of occurrences are deemed to represent the user’s main interests. If a twitter user does not share links/tweets, the twitter accounts they are following are profiled and the user is considered to have the collective interests of the users he/she is following.

The nanobird twitter application is able to directly spot the interests of every single twitter user and recommend other twitter users sharing common interests (Figure 21). The application is also able to filter out the non-relevant links that are coming up in a user’s feed. Massive amount of noise in users account could be turned into a fine tune or pitch of tweets/links that are relevant to the users’ interests.

Similarly, the nanobird social media application profiles users on other social networking sites (e.g. Facebook, Google+) and determines their interest by processing the links and text they share.

2.2 News

The nanobird news application’s main focus is on providing users with news tailored to their likings. The news application has three main components: a profiling component, a news context analyzer, a news delivery component.
Profiling component: Learning a user’s news interests could be achieved in two ways; either by conducting an initial profiling of the provided user’s social media accounts, or by learning, over time, from the news the user reads on the nanobird news application. The latter learning technique is based on monitoring a user’s reading behavior and some aging algorithms to clean out entities that are no longer relevant to a user and entities that were only part of a sudden world event or non-timeless event.

News context analyzer: As fresh news articles are published, the nRE profiles every new article and extracts the set of words that best describes the article.

News delivery component: By combining the results from the profiling component and the news context analyzer component, it becomes possible to deliver news closest to the users’ interests.

2.3 Movie Recommendations
The nanobird movie application is an application that provides movie recommendation to users. It comprises two main components: an movie understanding component and recommendation component.

Movie understanding component: Every film’s plot synopsis and summary is passed through the nRE where the relevant entities of the movie are extracted and classified.

Recommendations component: A user is asked to input a movie of their preference and the recommendation component provides him/her with the closest matching movie in terms of synopsis and/or genre such as “spy action thriller”. The nanobird movie application will always recommend more recent films (including the latest films out on DVD, in theaters, or coming soon) before the outdated ones.

2.4 Book Suggestions
The nanobird book recommendation application is similar to the nanobird movie recommendations one except that the profiling of a book is achieved by passing the story plot/summary to the nRE.

2.5 Advertisement
The nanobird advertisement application helps content providers increase their revenue from advertisements by displaying relevant ads on their published sites. Relevant ads are displayed based on the site-visitor’s interests (extracted from his/her social media accounts as previously described) or based on the content of the pages itself using nRE.

2.6 Search
By understanding text just as a human does, the nanobird search application is able to organize all the content of the web just as a human would organize documents in a ling cabinet based on subject. A nanobird search application would be able to give search results based on relevancy to what a user searches for. The nanobird search application could be a plug-in to search engines such as Google and Bing to improve ranking of results and results classification.
Figure 1. The nanobird Relevancy Engine Components Diagram
Figure 2. A subset of the unambiguous phrases dictionary

Figure 3. An example of ambiguous phrases (apple, atmosphere, Wednesday) and their mapping to unambiguous phrases
Figure 4. An example of phrase aliases and their mapping to unambiguous phrases
Figure 5. A fingerprint of the phrase “Apple Inc.”
This article is about the technology company. For other companies named "Apple", see Apple (disambiguation).

Apple Inc. (NASDAQ: AAPL), previously Apple Computer, Inc., is an American multinational corporation that designs and markets consumer electronics, computer software, and personal computers. The company's best-known hardware products include the Macintosh line of computers, the iPod, the iPhone, and the iPad. Apple software includes the Mac OS X operating system, the iTunes media browser, the iWork suite of productivity software, the Aperture, a professional photography package, Final Cut Studio, a suite of professional audio and film industry software products; Logic Studio, a suite of music production tools; the Safari web browser; and iOS, a mobile operating system for smartphones and tablets. As of October 2010, the company operates 317 retail stores in 14 countries and an online store where hardware and software products are sold. As of September 2011, Apple is the largest publicly traded company in the world by market capitalization and the largest technology company in the world by revenue and operating profit.

Established on April 1, 1976 in California, and incorporated January 3, 1977, the company was previously named Apple Computer, Inc., for its first 30 years, but removed the word "Computer" on January 9, 2007 to reflect the company's ongoing expansion into the consumer electronics market in addition to its traditional focus on personal computers. As of September 2010, Apple had 66,600 full-time employees and 2,500 temporary full-time employees worldwide and had worldwide annual sales of $53.2 billion.

For reasons as various as its philosophy of comprehensive aesthetic design to its distinctive advertising campaigns, Apple has established a unique reputation in the consumer electronics industry. This includes a customer base that is devoted to the company and its brand, particularly in the United States. Fortune magazine named Apple the most admired company in the United States in 2000 and in the world in 2009, 2010, and 2012. (The company has also received widespread criticism for its contractors' labor, environmental, and business practices.)

History

Main article: History of Apple Inc.

1976–1980: The early years

Apple was founded on April 1, 1976 by Steve Jobs, Steve Wozniak, and Ronald Wayne. To sell the Apple I personal computer kit, they were hand-built by Wozniak and first shown to the public at the Homebrew Computer Club (HCC). The Apple I was sold as a motherboard (with the CPU, RAM, and basic text-based video chip)—less than what is today considered a complete personal computer. The Apple II went on sale in July 1976 and was priced at $1,385 in 2011 dollars, adjusted for inflation.

Apple was incorporated January 3, 1977 without Wayne, who sold his share of the company back to Jobs and Wozniak for $1,600. Multi-millionaire Mike Markkula provided essential business expertise and funding of $250,000 during the incorporation of Apple.

The Apple II was introduced on April 10, 1977 at the first West Coast Computer Fair. It differed from its major rivals, the TRS-80 and Commodore PET, because it came with color graphics and an open architecture. While early models used ordinary cassette tapes as storage devices, they were superseded by the introduction of a 5½-inch floppy disk drive and interface, the Disk II.

The Apple II was chosen to be the desktop platform for the first "killer app" of the business world—the Visicalc spreadsheet program. VisiCalc created a business market for the Apple II, and gave home users an additional reason to buy an Apple II—compatibility with the office. According to Brian Bagnall, Apple exaggerated its sales figures and was a distant third place to Commodore and Tandy until VisiCalc came along.

By the end of the 1970s, Apple had a staff of computer designers and a production line. The company introduced the ill-fated Apple III in May 1980 in an attempt to compete with IBM and Microsoft in the business and corporate computing market.

Jobs and several other Apple engineers including Jeff Raskin visited Xerox PARC in December 1979 to see the Xerox Alto. Xerox granted Apple engineers three days of access to the PARC facilities in return for the option to buy 100,000 shares (800,000 split-adjusted shares) of Apple at the pre-IPO price of $10 a share. By the time Apple went public, it generated more revenue than any other Silicon Valley startup.
Figure 6. A Wikipedia article split into five defined sections

Figure 7. A Wikipedia article about “Apple Inc.”
Figure 8. A Wikipedia article about “iPhone”
A personal computer (PC) is any general-purpose computer whose size, capabilities, and original sales price make it useful for individuals, and which is intended to be operated directly by an end-user with no intervening computer operator. In contrast, the batch processing or time-sharing models allowed large expensive mainframe systems to be used by many people, usually at the same time. Large data processing systems require a skilled staff to operate efficiently.

Software applications for personal computers include, but are not limited to, word processing, spreadsheet, database, web browsers and e-mail client, digital media playback, games, and myriad personal productivity and special-purpose software applications. Modern personal computers often have connections to the Internet, allowing access to the World Wide Web and a wide range of other resources. Personal computers may be connected to a local area network (LAN), either by a cable or a wireless connection. A personal computer may be a desktop computer or a laptop, tablet PC, or a handheld PC.

While early PC owners usually had to write their own programs to do anything useful with the machines, today’s users have access to a wide range of commercial software and free software, which is provided in ready-to-run or require-compilation form. Since the early 1990s, Microsoft and Intel have dominated much of the personal computer market, first with MS-DOS and then with the Windows platform. Alternatives to Windows include Apple’s Mac OS X and the open-source Linux OSes. AMD is the major alternative to Intel. Applications and games for PCs are typically developed and distributed independently from the hardware and software vendors, whereas software for many mobile phones and other portable systems is approved and distributed through a centralized online store.

In July and August 2011, marketing businesses and journalists began to talk about the ‘Post-PC Era’, in which the desktop form factor was being replaced with more portable computing such as netbooks, tablet PCs, and smartphones.
Figure 10. A Wikipedia article about “Computer software”
Figure 11. A Wikipedia article about “Steve Jobs”
Given:
D denotes the set of dictionaries \( (D_U, D_M, D_L) \)
p denotes a phrase
G is an inputted text consisting of multiple words
S is a set of longest matching phrases such that \( S \subseteq D \)
\( \oplus \) is the operation of joining two phrases using whitespace

Input:
\[ G = w_1 \oplus w_2 \oplus \ldots \oplus w_n \]

Algorithm:
\[ g \leftarrow \text{split } G \text{ into set of words} \]
p \( \leftarrow \) \( \emptyset \)
S \( \leftarrow \) \( \emptyset \)
for \( i = 1 \) to \( n \)
\[ p \leftarrow p \oplus w_i \]
if \( s \) is a match in \( D \) then
\[ \text{Psaved} \leftarrow p \]
\[ i_{\text{saved}} \leftarrow i \]
continue
else if \( p \) is a submatch in \( D \) then
continue
else if \( \text{Psaved} \neq \emptyset \) then
\[ S \leftarrow S \cup \{ \text{Psaved} \} \]
\[ p \leftarrow \emptyset \]
\[ \text{Psaved} \leftarrow \emptyset \]
\[ i_{\text{saved}} \leftarrow i_{\text{saved}} + 1 \]

Figure 12. nanobird Phrase Tokenizer Algorithm
Figure 13. nanobird Relevancy Engine Flowchart
The Great Lakes are in Danger

The effects of the climate crisis are now damaging the Great Lakes:

"Some of the Great Lakes' treasured national parks are showing ill effects of climate change that are likely to worsen in coming decades, from shoreline erosion to decline of certain wildlife and plant species, a former park system administrator said Wednesday."

"Without changes in public policies and personal habits that pump greenhouse gases into the atmosphere, the parks could lose qualities that attract visitors and support unique ecosystems. Stephen Saunders, former deputy assistant secretary of the Interior Department, said in a report released by two advocacy groups."

Source: AP
Figure 15. Partial fingerprint of the phrase “Atmosphere of Earth”
Figure 16. Partial fingerprint of the phrase “Climate change”
Figure 17. Partial fingerprint of the phrase “Ecosystem”
Figure 18. Partial fingerprint of the phrase “Global warming”
Figure 19. Partial fingerprint of the phrase “Greenhouse gas”
Figure 20. An example of a nanobird Context Analyzer Graph
Figure 21. Example showing the interest of some Twitter users after being processed by the twitter application using nRE.