April 18, 2016

NAME NORMALIZATION

Vladimir Yakunin

David Talbot

NAME NORMALIZATION

Background

[0001] In many languages (particularly Slavic, Turkic and Finno-Ugric) names can change form (e.g., spelling) based on a context in which the names are used. For example, in Czech, the name “Novák” can become “Nováka”, “Nováku”, “Novákem”, etc., depending on whether it is used as the subject, object, or prepositional object in a sentence. This variation may pose a challenge for computer systems designed to identify a person's name in some text and/or to resolve the person’s name against a knowledge base (e.g. a dictionary or contact list) because most such databases typically only contain names in a subject form.

[0002] Stemming techniques such as the Porter stemming algorithm may attempt to stem words by stripping off suffixes. However these components are not typically trainable or able to learn a set of string edits that would be useful for normalizing names in a specific language. They also are not typically able to take advantage of contextual information to choose between multiple candidate normalizations for a single name. For instance, the name "Alexandru" could be either the accusative form of "Alexandra" or dative form of "Alexandr" – female and male first names respectively – in Czech.

Summary

[0003] This disclosure is directed generally to methods and apparatus for normalizing tokens found in text input prior to attempting to match those tokens to a database of names (e.g., a contact list). In various implementations, a set of suffix tuples, which may be optionally annotated, may be used to normalize tokens found in text obtained from, for instance, a user speaking into a microphone of a mobile phone or smart watch. For example, a suffix tuple, <-u, -a, “feminine”> may indicate that the suffix “u” may be replaced by the suffix “a” to normalize a name, and that the resulting normalized name should be classified and/or annotated as “feminine.” Thus, for example, the spoken name “Annu” may be changed to “Anna” and labelled as “feminine.”

Brief Description of the Drawings

[0004] Fig. 1 illustrates an example environment in which tokens from segments of text may be normalized, in accordance with various implementations.

[0005] Fig. 2 illustrates one example of how spoken text may be normalized for a dialer app.

[0006] Fig. 3 is a flow chart illustrating an example method of normalizing tokens to generate and apply candidate normalized names.

[0007] Fig. 4 illustrates an example architecture of a computer system.

Detailed Description

[0008] Fig. 1 illustrates an example environment in which one or more tokens of textual input may be normalized to generate one or more candidate normalized names, in accordance with various implementations. The example environment includes a client device 106 and a knowledge system 102. Knowledge system 102 may be implemented in one or more computers that communicate, for example, through a network (not depicted). Knowledge system 102 is an example of a system in which the selected aspects of systems, components, and techniques described herein may be implemented and/or with which systems, components, and techniques described herein may interface. Additionally or alternatively, selected aspects of systems, components, and techniques described herein may also be implemented on client device 106.

[0009] A user may interact with knowledge system 102 via client device 106 and/or other computing systems (not shown). Client device 106 may be a computer coupled to the knowledge system 102 through one or more networks 110 such as a local area network (LAN) or wide area network (WAN) such as the Internet. The client device 106 may be, for example, a desktop computing device, a laptop computing device, a tablet computing device, a mobile phone computing device, a computing device of a vehicle of the user (*e.g.*, an in-vehicle communications system, an in-vehicle entertainment system, an in-vehicle navigation system), or a wearable apparatus of the user that includes a computing device (*e.g.*, a watch of the user having a computing device, glasses of the user having a computing device). Additional and/or alternative client devices may be provided. While the user likely will operate a plurality of computing devices, for the sake of brevity, examples described in this disclosure will focus on the user operating client device 106.

[0010] Client device 106 may operate one or more applications and/or components, including but not limited to a microphone 107, a telephone dialer application 109, a position coordinate component, such as a global positioning system ("GPS") component 111, and so forth. In some instances, one or more of these applications and/or components may be operated on multiple client devices operated by the user. Other components of client device 106 not depicted in Fig. 1 that may provide signals include but are not limited to barometers, Geiger counters, cameras, light sensors, presence sensors, thermometers, health sensors (e.g., heart rate monitor, glucose meter, blood pressure reader), accelerometers, gyroscopes, and so forth.

[0011] Client device 106 and knowledge system 102 each include one or more memories for storage of data and software applications, one or more processors for accessing data and executing applications, and other components that facilitate communication over a network. The operations performed by client device 106 and/or knowledge system 102 may be distributed across multiple computer systems. Knowledge system 102 may be implemented as, for example, computer programs running on one or more computers in one or more locations that are coupled to each other through a network.

[0012] In various embodiments, client device 106 may provide segments of text in various forms to knowledge system 102. Knowledge system 102 may analyze the text segments, normalize one or more tokens identified as names that are contained therein, and generate one or more candidate normalized names. Knowledge system 102 may provide the candidate normalized name(s) to client device 106. The candidate normalized name(s) may then be matched, e.g., by knowledge system 102 or client device 106, to a list of names such as a contact database. For example, client device 106 may then match the candidate normalized names to a name in a contact list available to dialer 109, and may dial a telephone number associated with that contact.

[0013] The text segment provided by client device 106 to knowledge system 102 may take various forms and may come from various sources. In some implementations, client device 106 may provide text from an email, text, social networking status, or other electronic document to knowledge system 106. Additionally or alternatively, client device 106 may be configured to sense one or more audible sounds, e.g., using microphone 107, and may provide audio information based on the sensed one or more audible sounds to various other components. The audio information may include a digital recording of the audio and/or a transcription of the audio. Those other components, examples of which are described in more detail below, may perform various operations based on the received text segment to identify one or more tokens, and to perform techniques described herein to normalize one or more of those tokens so that the normalized tokens may be used to generate one or more candidate normalized names.

[0014] In some implementations, the text provided by client device 106 to knowledge system 102 may be annotated in various ways, e.g., by client device 106 and/or components of knowledge system 102. For example, tokens of the text may be identified and annotated as particular parts of speech, nouns, verbs, etc. Various components of knowledge system 102 may analyze the text and/or annotations to attempt to normalize one or more of the identified tokens into a normalized name.

[0015] In various implementations, knowledge system 102 may include a language identifier engine 122 connected to a first index 123 and a normalization engine 124 connected to a second index 125. In some implementations one or more of engines 122 and 124 may be omitted. In some implementations all or aspects of one or more of engines 122 and 124 may be combined. In some implementations, one or more of engines 122 and 124 may be implemented in a component that is separate from knowledge system 102. In some implementations, one or more of engines 122 and 124, or any operative portion thereof, may be implemented in a component that is executed by client device 106.

[0016] Language identifier engine 122 may be configured to analyze text (and in some instances, annotations of the text) received from client device 106 to determine a language of the text. In some implementations, language identifier engine 122 may compare one or more tokens in the received text to one or more tokens contained in in multiple dictionaries of various languages, e.g., stored in index 123. Whichever dictionary is matched to the most tokens, the language of that dictionary may be identified as the language of the text.

[0017] In some implementations, language identifier engine 122 may identify language of the received text using a variety of contextual signals independent from the text received from client device 106. For example, in some embodiments, position coordinates from GPS unit 111 may be indicative of a location of a user of client device 106, which in turn may be used as a clue to which language the user is using. For example, suppose position coordinates provided by GPS unit 111 indicate that client device is in the Czech Republic. Language identifier engine 122 may take that fact into account when determining a language of text received from client device 106. For example, language identifier engine 122 may be more likely to identify the language as Czech.

[0018] Normalization engine 124 may be configured to utilize one or more models stored in index 125 to analyze tokens contained in a text segment that are "non-normalized" (e.g., inflected) forms of names in order to generate one or more candidate normalized names. Index 125 may store various types of models. In some implementations, index 125 may store a suffix tree. Each node of the suffix tree may include a suffix that may be found in a token and a potential edit that may be applied to the token's suffix in order to potentially "normalize" the token. Each node optionally may include a label such as "feminine," "masculine," "accusative," "dative," "nominative," and so forth. Many suffixes may be represented by multiple nodes, with one node corresponding to a potential edit of the suffix in a first language, another node corresponding to potential edit of the suffix in a second language, and so on. In some implementations, each node may have a running count (or "edit frequency") of how many times it is successfully used to normalize a name. This running count may be used, for instance, to calculate a score for a candidate normalized name generated using the node.

[0019] Suppose the token "Alexandru" is identified within a segment of text. In Czech, "Alexandru" could be used as the accusative form of "Alexandra" or the dative form of "Alexandr." Normalization engine 124 may traverse a Czech suffix tree to identify nodes corresponding to the suffix "u". There may be at least two such nodes identified: one labeled "feminine" that includes a potential edit that replaces the suffix "u" with "a"; and another labeled “masculine” that includes a potential edit to replace the suffix “u” with “”. Each of these nodes may be used by normalization engine to generate a candidate normalized name, in this instances “Alexandra” and “Alexandr.”

[0020] In various implementations, normalization engine 124 may additionally be configured to calculate and assign scores to candidate normalized names. The scores may be indicative of how reliable the edits to the original tokens are believed to be. In some implementations, a score may be assigned to a candidate normalized name based on a length of a suffix that was replaced in the original token and/or a length of the edit suffix that was used as a replacement.

[0021] In some implementations, a score may be assigned to a candidate normalized name based on the aforementioned “edit-frequency” associated with the edit. For example, if a first replacement suffix found in a first candidate normalized name is used as an edit more frequently than a second replacement suffix found in a second candidate normalized name, the first candidate normalized name may be assigned a higher score than the second candidate normalized name.

[0022] Additionally or alternatively, in some implementations, a context in which the original token was used may be considered in assigning a score to a candidate normalized name generated from the original token. Suppose a textual segment in which an original token was identified was generated by a user operating a client device in a particular country (e.g., as might be determined from position coordinates provided by GPS unit 111) that has an official language. Suppose further that a suffix tree stored in index 125 and utilized by normalization engine 124 contains nodes for multiple languages. An original suffix of that token may match multiple suffix nodes in a suffix tree, and thus multiple candidate normalized names may be generated. However, perhaps only some of those multiple matching suffix nodes are associated with the official language of the country in which the user was located when the text segment was generated. Candidate normalized names generated from those matching suffix nodes that are associated with the official language may be scored higher than candidate normalized names generated from other suffix nodes associated with other languages.

[0023] In this specification, the term "database" and "index" will be used broadly to refer to any collection of data. The data of the database and/or the index does not need to be structured in any particular way and it can be stored on storage devices in one or more geographic locations. Thus, for example, the indices 123, and/or 125 may include multiple collections of data, each of which may be organized and accessed differently.

[0024] Fig. 2 schematically depicts one example of how audible speech 250 may be captured and processed to generate one or more candidate normalized names, as well as how those candidate normalized names may be matched to a name list on client device 106, in accordance with various implementations. In Fig. 2, speech 250 is first detected by microphone 107 of client device 106. However, this is not meant to be limiting. Speech 250 may be captured by any pressure wave sensor (e.g., acoustic-to-electric transducer) operated by or associated with a user, such as a microphone in the user's home, office, and/or car. Additionally, the pressure wave sensor need not be a standalone microphone. In various implementations, it may be integral with a smart phone, with a video camera (e.g., a security camera), and so forth. More generally, text segments that are analyzed for non-normalized forms of names need not be captured by a microphone. As noted above, any segment of text, such as from an email, text message, online article, electronic book, etc., may be analyzed using disclosed techniques.

[0025] In some implementations, language identifier engine 122 may analyze the segment of text using various techniques, such as those described above, to identify a language of the segment of text. Language identifier engine 122 may provide a designation of the identified language and the segment of text (e.g., in raw form or separated into tokens) to normalization engine 124. If normalization engine 124 uses a universal or language-independent model to normalize names, normalization engine 124 may use the language designation to assisting in scoring candidate normalized names. If normalization engine 124 uses multiple language-specific models, normalization engine 124 may use the language designation to select which models to utilize. In other implementations, language identifier engine 122 may be omitted altogether.

[0026] As described above, normalization engine 124 may apply one or more models, such as one or more suffix trees, to generate one or more candidate normalized names.

Normalization engine 124 may also calculate and assign scores to the candidates. As will be described below, in some implementations, normalization engine 124 may cull candidate normalized names, e.g., discarding or otherwise disregarding those with scores that fail to satisfy a threshold, etc. The remaining one or more candidate normalized names may be passed back to client device 106.

[0027] Client device 106 may match the one or more candidate normalized names to one or more names in a list of names, e.g., in a contact list stored on client device. For example, in Fig. 2, dialer app 109 receives the candidate normalized names and matches them to contacts in the user's contact list. Assuming there is a single matching contact, dialer may dial the phone number associated with that contact. If there are multiple matches, the "best" match may be selected (e.g., the contact that is most similar to the candidate normalized name), or all potentially matching contacts may be presented to the user, and the user can select which contact should be dialed. If multiple candidate normalized names match multiple contacts, then in some embodiments, edit frequencies associated with underlying edits that were used to create the candidate normalized names may be used to select which candidate should be selected. In some implementations, application-specific rules/scoring functions may also be applied to select which candidate normalized name should be used.

[0028] In another aspect, models used by normalization engine 124 may be trained in various ways. In some implementations, a training algorithm may obtain pairs of names in subject (e.g., normalized/dictionary form) and inflected (e.g., non-normalized) form. In some implementations, the pairs may be associated with one or more labels (e.g. "feminine" or "masculine"). The training algorithm may compare these strings to determine a common prefix and/or a difference between suffixes. The suffix difference may constitute a potential edit that may be made to the inflected form of the name in order to "normalize" it into its subject form. For example, given the subject form of the name "Anna" and the non-subject (e.g., inflected) form "Annu," the training algorithm may derive a string edit "u"=> "a". The string edit operation may then be stored in one or more models (e.g., added as nodes to a suffix tree) to be used by normalization engine 124 in the future. In some embodiments, the edits may be sorted within a model by frequency such that more common edits appear first in the model. If any labels were included with the training pair, these labels may be associated with the string edits.

[0029] In some implementations, a list of names in normalized (subject) form and a list of templates that each have a slot that requires a name in a specific form may be provided to native speakers of one or more languages. The native speakers may be requested to adjust the normalized names to fit into each template in turn. For example, a template may include a list of names: “Anna”, “Ivan”, “Bertrand”, etc., and a list of sentences with a blank that can be filled with a name. For example, the following is one non-limiting example of a template that may be filled in by native speakers:

Call [ ]
[ ] was born in France.
I saw [ ]

[0030] A native speaker of the language in question will take the names from the list and produce the correct form for each template. For example, for “Anna”:

Call [Anne]
[Anna] was born in France.
I saw [Annu]

As another example, the template may be filled out as follows for “Ivan”:

Call [Ivanu]
[Ivan] was born in France.
I saw [Ivana]

Each template and name pair can produce a tuple of the form (Anna, Annu), (Ivan, Ivana) etc. that serves as input training data for the training algorithm mentioned above.

[0031] Fig. 3 schematically depicts an example method 300 of normalizing one or more tokens of a segment of text and generating one or more candidate normalized names, in accordance with various implementations. For convenience, the operations of the flow chart are described with reference to a system that performs the operations. This system may include various components of various computer systems. For instance, some operations may be performed at the client device 106, while other operations may be performed by one or more components of the knowledge system 102. Moreover, while operations of method 300 are shown in a particular order, this is not meant to be limiting. One or more operations may be reordered, omitted or added.

[0032] At block 302, the system may receive, e.g., from client device 106, a text segment. As noted above, this text segment may come in various forms, such a transcription of an audio sample (e.g., obtained from microphone 107), from one or more textual documents (e.g., emails, letters, text messages, web pages, online news articles, contracts, etc.), and so forth. At block 304, the system may identify one or more tokens within the text segment. For example, tokens may come in the forms of words that are separated from other tokens/works by white space.

[0033] At block 306, the system may traverse the aforementioned suffix tree to locate a suffix associated with each token identified at block 304. As noted above, each suffix node may contain one or more edits that may be applied to possibly obtain a normalized version of the token. At block 308, matching edits (i.e. those edits contained in nodes having suffixes matching the tokens) may be applied to generate candidate normalized names.

[0034] At block 310, the system may assign scores to candidate normalized names based on a variety of signals, as was described above. At block 312, the system may discard or otherwise disregard all but a predetermined number of, or a predetermined percentage of, the candidate normalized names generated at block 308. The number/percentage of candidate normalized names that are not discarded or otherwise disregarded maybe selected based on a variety of factors. For example, in some implementations, a desired precision may be weighed against a desired recall measurement to determine how many candidate normalized names should be discarded/disregarded.

[0035] At block 314, the remaining candidate normalized names may be matched against a particular list of names. For example, client device 106 may include or have access to a contact list (e.g., in association with dialer application 109).

[0036] Fig. 4 is a block diagram of an example computer system 410. Computer system 410 typically includes at least one processor 414 which communicates with a number of peripheral devices via bus subsystem 412. These peripheral devices may include a storage subsystem 424, including, for example, a memory subsystem 425 and a file storage subsystem 426, user interface output devices 420, user interface input devices 422, and a network interface subsystem 416. The input and output devices allow user interaction with computer system 410. Network interface subsystem 416 provides an interface to outside networks and is coupled to corresponding interface devices in other computer systems.

[0037] User interface input devices 422 may include a keyboard, pointing devices such as a mouse, trackball, touchpad, or graphics tablet, a scanner, a touchscreen incorporated into the display, audio input devices such as voice recognition systems, microphones, and/or other types of input devices. In general, use of the term "input device" is intended to include all possible types of devices and ways to input information into computer system 410 or onto a communication network.

[0038] User interface output devices 420, which in some implementations may correspond to output device 109 in Fig. 1, may include a display subsystem, a printer, a fax machine, or non-visual displays such as audio output devices. The display subsystem may include a cathode ray tube (CRT), a flat-panel device such as a liquid crystal display (LCD), a projection device, or some other mechanism for creating a visible image. The display subsystem may also provide non-visual display such as via audio output devices. In general, use of the term "output device" is intended to include all possible types of devices and ways to output information from computer system 410 to the user or to another machine or computer system.

[0039] Storage subsystem 424 stores programming and data constructs that provide the functionality of some or all of the modules described herein. For example, the storage subsystem 424 may include the logic to perform selected aspects of method 300, as well as one or more of the operations performed by language identifier engine 122, normalization engine 124, and so forth.

[0040] These software modules are generally executed by processor 414 alone or in combination with other processors. Memory 425 used in the storage subsystem 424 can include a number of memories including a main random access memory (RAM) 430 for storage of instructions and data during program execution and a read only memory (ROM) 432 in which fixed instructions are stored. A file storage subsystem 426 can provide persistent storage for program and data files, and may include a hard disk drive, a floppy disk drive along with associated removable media, a CD-ROM drive, an optical drive, or removable media cartridges. The modules implementing the functionality of certain implementations may be stored by file storage subsystem 426 in the storage subsystem 424, or in other machines accessible by the processor(s) 414.

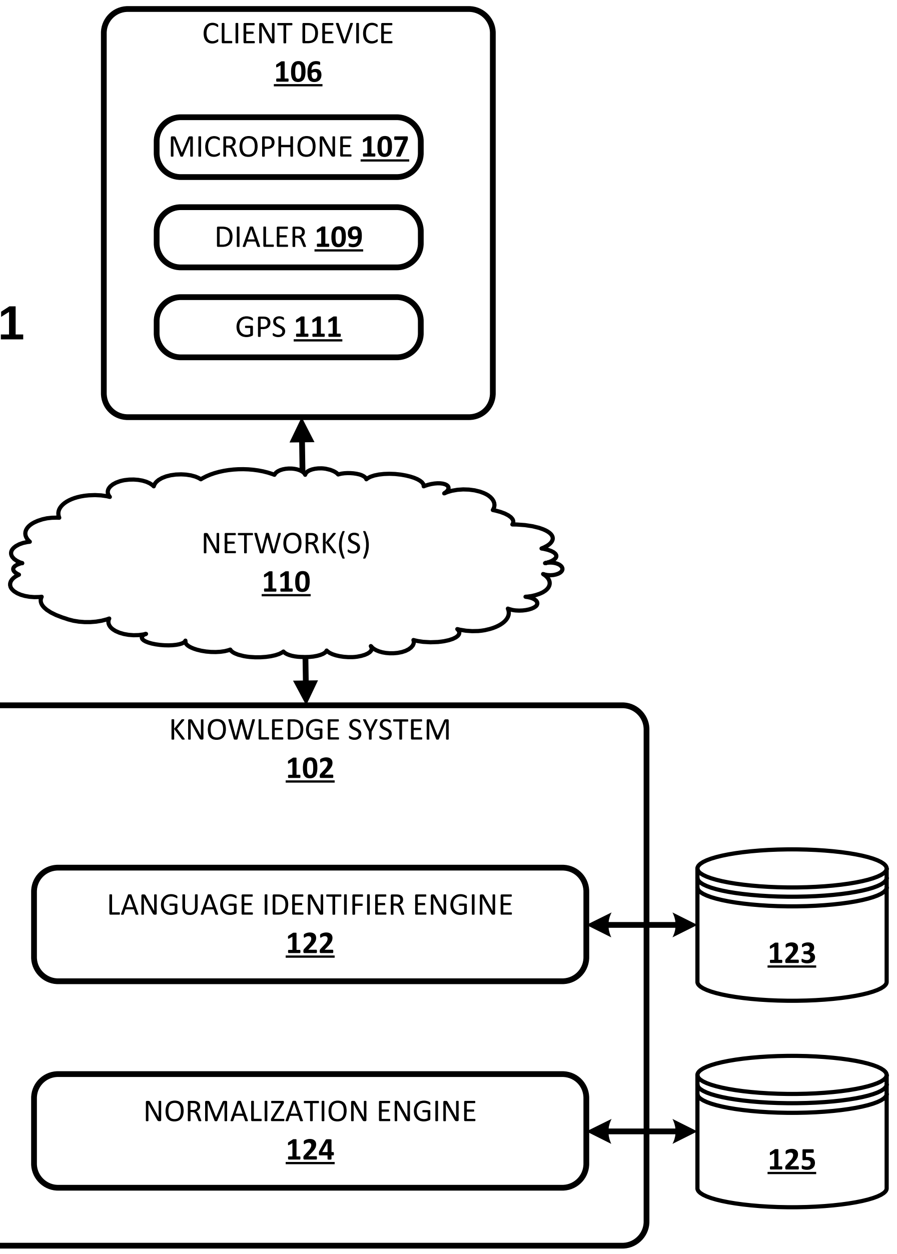
[0041] Bus subsystem 412 provides a mechanism for letting the various components and subsystems of computer system 410 communicate with each other as intended. Although bus subsystem 412 is shown schematically as a single bus, alternative implementations of the bus subsystem may use multiple busses.

[0042] Computer system 410 can be of varying types including a workstation, server, computing cluster, blade server, server farm, or any other data processing system or computing device. Due to the ever-changing nature of computers and networks, the description of computer system 410 depicted in Fig. 4 is intended only as a specific example for purposes of illustrating some implementations. Many other configurations of computer system 410 are possible having more or fewer components than the computer system depicted in Fig. 4.

[0043] In situations in which the systems described herein collect personal information about users, or may make use of personal information, the users may be provided with an opportunity to control whether programs or features collect user information (e.g., information about a user's social network, social actions or activities, profession, a user's preferences, or a user's current geographic location), or to control whether and/or how to receive content from the content server that may be more relevant to the user. Also, certain data may be treated in one or more ways before it is stored or used, so that personal identifiable information is removed. For example, a user's identity may be treated so that no personal identifiable information can be determined for the user, or a user's geographic location may be generalized where geographic location information is obtained (such as to a city, ZIP code, or state level), so that a particular geographic location of a user cannot be determined. Thus, the user may have control over how information is collected about the user and/or used.

[0044] While several implementations have been described and illustrated herein, a variety of other means and/or structures for performing the function and/or obtaining the results and/or one or more of the advantages described herein may be utilized, and each of such variations and/or modifications is deemed to be within the scope of the implementations described herein. More generally, all parameters, dimensions, materials, and configurations described herein are meant to be exemplary and that the actual parameters, dimensions, materials, and/or configurations will depend upon the specific application or applications for which the teachings is/are used. Those skilled in the art will recognize, or be able to ascertain using no more than routine experimentation, many equivalents to the specific implementations described herein. It is, therefore, to be understood that the foregoing implementations are presented by way of example only. Implementations of the present disclosure are directed to each individual feature, system, article, material, kit, and/or method described herein. In addition, any combination of two or more such features, systems, articles, materials, kits, and/or methods, if such features, systems, articles, materials, kits, and/or methods are not mutually inconsistent, is included within the scope of the present disclosure.

Fig. 1


CLIENT DEVICE
106
MICROPHONE 107
DIALER 109
GPS 111
NETWORK(S)
110
KNOWLEDGE SYSTEM
102
LANGUAGE IDENTIFIER ENGINE
122
123
NORMALIZATION ENGINE
124
125

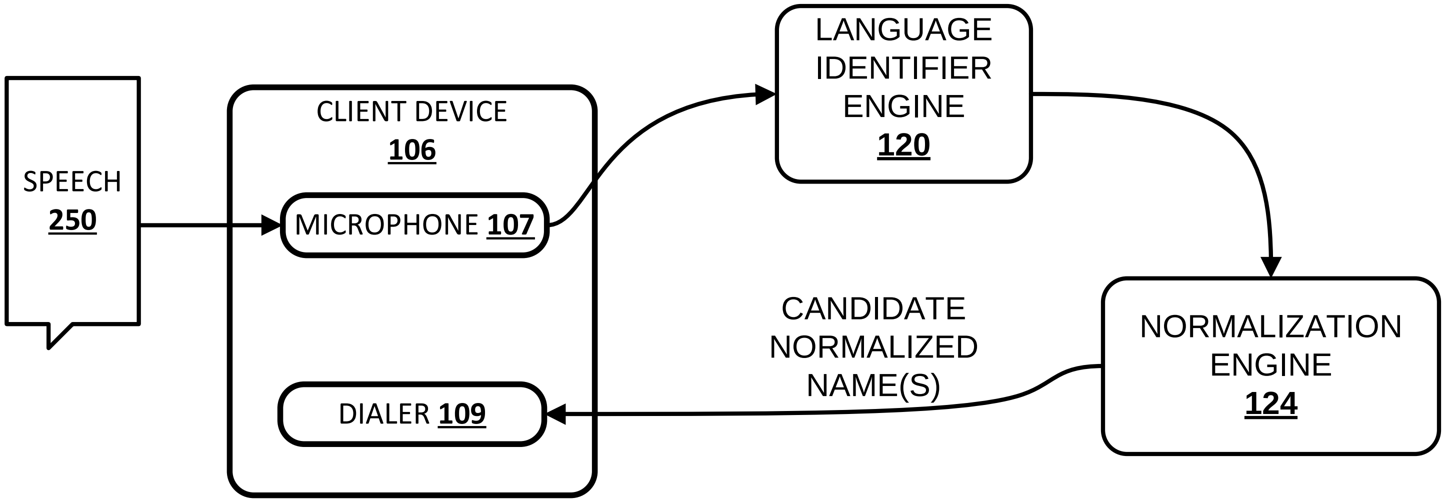

SPEECH
250
CLIENT DEVICE
106
MICROPHONE 107
DIALER 109
LANGUAGE
IDENTIFIER
ENGINE
120
NORMALIZATION
ENGINE
124
CANDIDATE
NORMALIZED
NAME(S)

Fig. 2

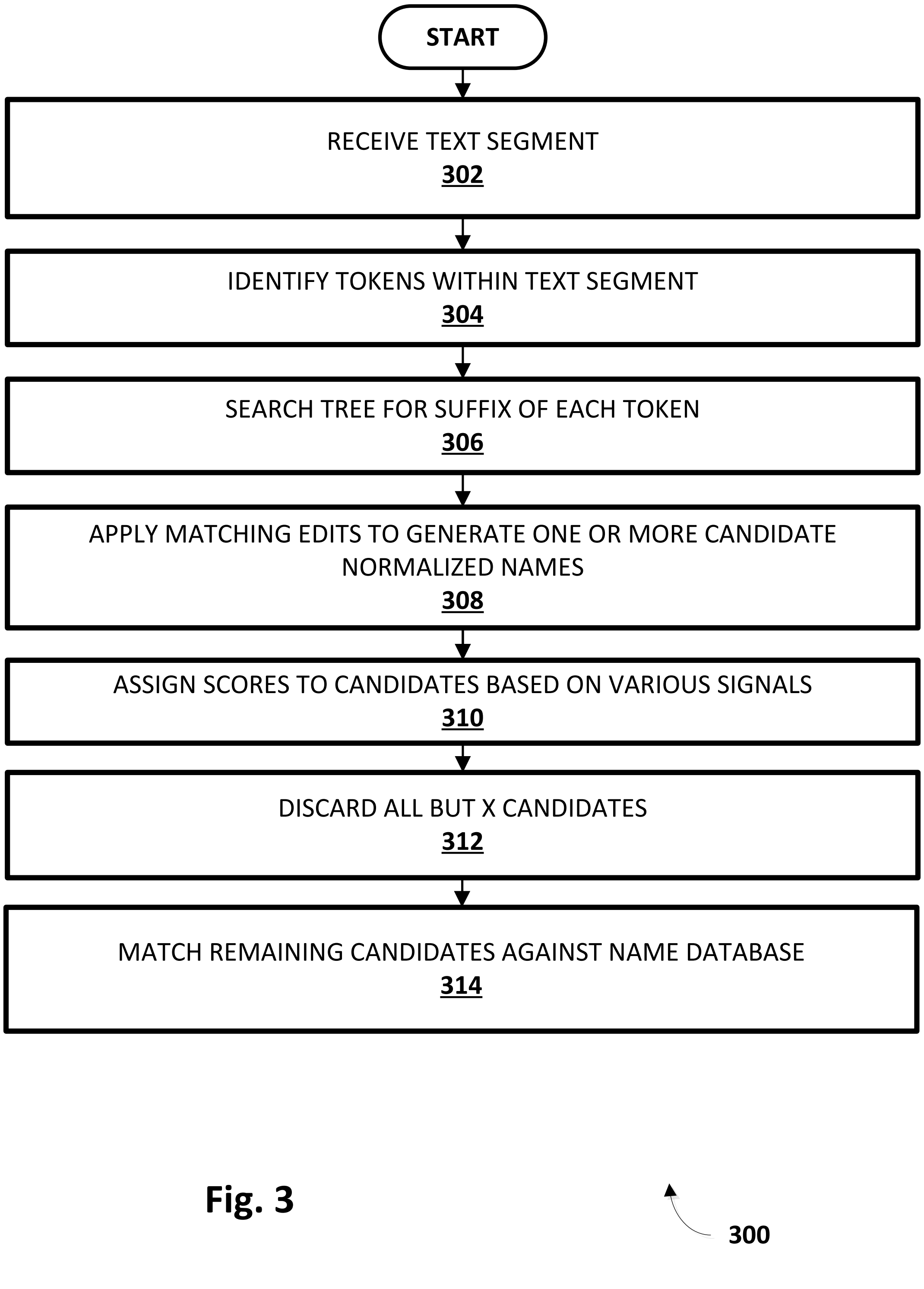

START
RECEIVE TEXT SEGMENT
302
IDENTIFY TOKENS WITHIN TEXT SEGMENT
304
SEARCH TREE FOR SUFFIX OF EACH TOKEN
306
APPLY MATCHING EDITS TO GENERATE ONE OR MORE CANDIDATE NORMALIZED NAMES
308
ASSIGN SCORES TO CANDIDATES BASED ON VARIOUS SIGNALS
310
DISCARD ALL BUT X CANDIDATES
312
MATCH REMAINING CANDIDATES AGAINST NAME DATABASE
314
300

Fig. 3

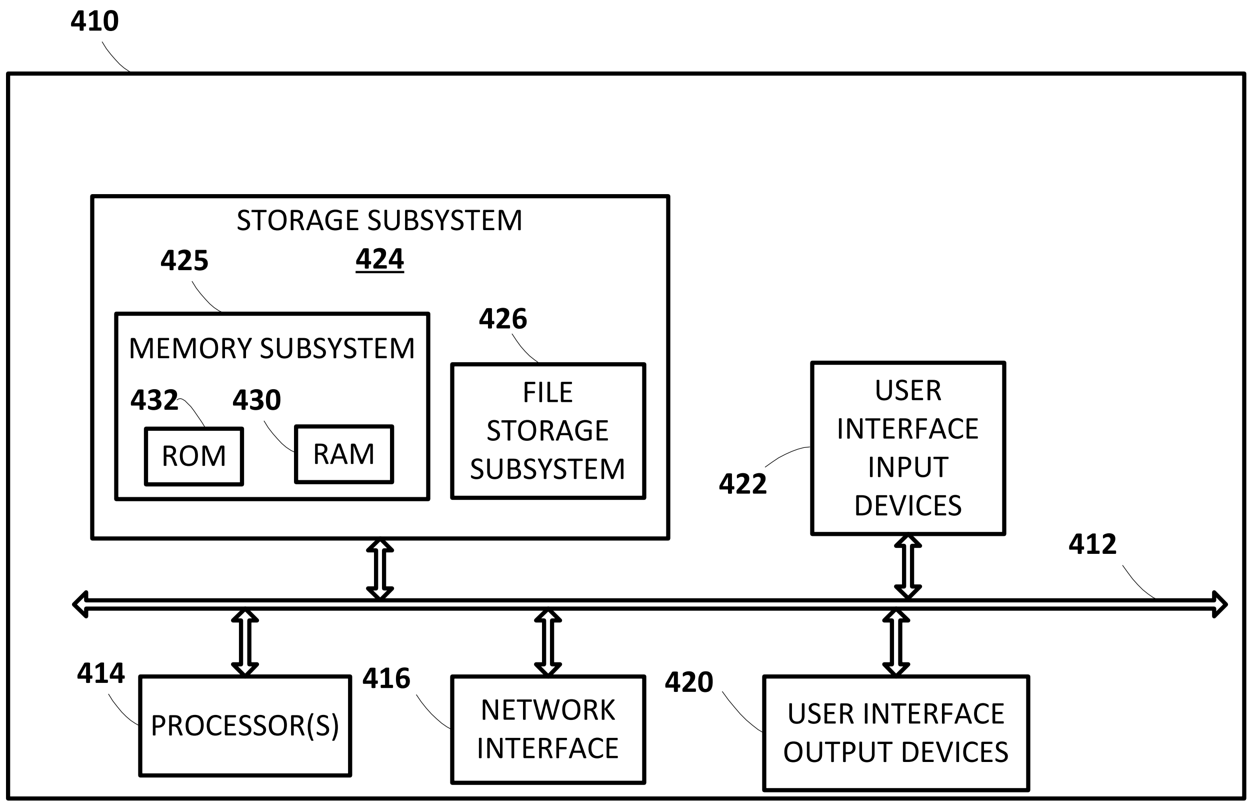

410
STORAGE SUBSYSTEM
424
425
MEMORY SUBSYSTEM
432
430
ROM
RAM
426
FILE STORAGE SUBSYSTEM
USER INTERFACE INPUT DEVICES
422
412
414
PROCESSOR(S)
416
NETWORK INTERFACE
420
USER INTERFACE OUTPUT DEVICES

Fig. 4