METHODS, APPARATUSES, AND DEVICES FOR GENERATING WORD VECTORS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This patent application claims priority to Chinese Patent Application No. 201810113710.3, filed February 5, 2018, and entitled "METHODS, APPARATUSES, AND DEVICES FOR GENERATING WORD VECTORS ", which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] The present specification relates to the technical field of computer software, and in particular, to methods, apparatuses, and devices for generating word vectors.

BACKGROUND

[0003] Most of today's natural language processing solutions employ a neural network-based architecture in which an important underlying technology is word vector. A word vector is a vector that maps a word to a fixed number of dimensions, where the vector represents the semantic information of the word.

[0004] In existing technologies, commonly used algorithms for generating word vectors include, for example, Google's word vector algorithm, Microsoft's deep neural network algorithm, etc.

[0005] In view of the existing technologies, there is a need for more accurate solutions for generating word vectors.

SUMMARY

[0006] Implementations of the present specification provide methods, apparatuses, and devices for generating word vectors, to provide a more reliable solution for generating word vectors.

[0007] The implementations of the present specification provide the following solutions:

[0008] An implementation of the present specification provides a method for generating word vectors, including: obtaining individual words by segmenting a corpus; establishing a feature vector for each obtained word based on n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word corresponding to the n-ary character; training a recurrent neural network based on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus; and generating a word vector for each obtained word based on the feature vector of the obtained word and the trained recurrent neural network.

[0009] An implementation of the present specification provides an apparatus for generating word vectors, including: an acquisition module, configured to obtain individual words by segmenting a corpus; an establishment module, configured to establish a feature vector for each obtained word based on n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word corresponding to the n-ary character; a training module, configured to train a recurrent neural network based on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus; and a generation module, configured to generate a word vector for each obtained word based on the feature vector of the obtained word and the trained recurrent neural network.

[0010] An implementation of the present specification provides another method for generating word vectors, including:

Step 1: Establishing a vocabulary of words obtained by segmenting a corpus, where the obtained words exclude word(s) that appear for less than a predetermined number of times in the corpus.

Step 2: Determining a total number of n-ary characters corresponding to all of the obtained words, where same n-ary characters are counted once, and each n-ary character represents n consecutive characters of an obtained word corresponding to the n-ary character.

Step 3: Establishing, for each obtained word, based on each n-ary character corresponding to the obtained word, a feature vector whose dimensionality is the total number, where each dimension of the feature vector corresponds to a different n-ary character, and the value of each dimension indicates whether a corresponding n-ary

character is mapped to the obtained word corresponding to the feature vector.

Step 4: Traversing the corpus on which word segmentation is performed, and performing step 5 with respect to a current word accessed during the traversal, and if the traversal is completed, performing step 6; otherwise continuing the traversal.

Step 5: Using the current word as a center, sliding towards both sides for at most k words to establish a window, and using the words in the window except the current word as context words, and inputting a sequence formed by feature vectors of all the context words into a sequence representation layer of a recurrent neural network for recurrent calculation to obtain a first vector; inputting a feature vector of the current word and a feature vector of a negative sample word selected in the corpus into a full connection layer of the recurrent neural network for calculation to obtain a second vector and a third vector; and updating parameters of the recurrent neural network based on the first vector, the second vector, the third vector, and a specified loss function.

The recurrent calculation is performed according to the following formula:

$$s_t = \sigma\left(Ux_t + Ws_{t-1}\right)$$

$$o_t = softmax\left(Vs_t\right)$$

The loss function includes:

$$l\left(w,c;U,W,V,\varsigma,\tau\right) = log\left(1+\sum_{m=1}^{\lambda} exp\left(-\gamma\cdot\left(s\left(w,c\right)-s\left(w_m^{'},c\right)\right)\right)\right)$$

where x_t indicates an input unit of the sequence representation layer at the time t, that is, a feature vector of the t+1 $^{\text{th}}$ context word associated with the current word; s_t indicates a hidden unit of the sequence representation layer at the time t; o_t indicates a vector obtained through recurrent calculation of the feature vectors of the first t+1 context words associated with the current word; U, W, and V indicate weight parameters of the sequence representation layer; σ indicates an activation function; c indicates the first vector; w indicates the second vector; $w_m^{'}$ indicates the third vector corresponding to the m $^{\text{th}}$ negative sample word; ς indicates a weight parameter of the full connection layer; τ indicates an offset parameter of the full connection layer; γ indicates a hyperparameter; s indicates a similarity calculation function; and λ indicates the number of negative sample words.

Step 6: Inputting the feature vector of each obtained word into the full connection layer of the trained recurrent neural network for calculation to obtain corresponding word vectors.

[0011] An implementation of the present specification provides a device for generating word vectors, including: at least one processor, and a memory communicatively connected to the at least one processor, where the memory stores instructions that can be executed by the at least one processor, and the instructions are executed by the at least one processor to enable the at least one processor to: obtain individual words by segmenting a corpus; establish a feature vector for each obtained word based on n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word corresponding to the n-ary character; train a recurrent neural network based on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus; and generate a word vector for each obtained word based on the feature vector of the obtained word and the trained recurrent neural network.

[0012] The previously described at least one technical solution adopted in the implementations of the present specification can achieve the following beneficial effects: The recurrent neural network can depict the context and overall semantic information of the word through the recurrent calculation, and extract more context and semantic information, and the n-ary character can express the word on more finely, so that the word vector can be more accurately generated.

BRIEF DESCRIPTION OF DRAWINGS

[0013] To describe the technical solutions in the implementations of the present specification or in the existing technologies more clearly, the following is a brief introduction of the accompanying drawings for illustrating such technical solutions. Clearly, the accompanying drawings described below are merely some implementations of the present specification, and a person skilled in the art can derive other drawings from such accompanying drawings without making innovative efforts.

[0014] FIG. 1 is a schematic diagram illustrating an overall architecture involved in an actual application scenario of solutions of the present specification;

[0015] FIG. 2 is a schematic flowchart illustrating a method for generating word vectors, according to some implementations of the present specification;

[0016] FIG. 3 is a schematic diagram illustrating a feature vector of an English word in an application scenario, according to some implementations of the present specification;

[0017] FIG. 4 is a schematic flowchart illustrating a principle of a method for generating word vectors based on a recurrent neural network in an actual application scenario, according to some implementations of the present specification;

[0018] FIG. 5 is a schematic structural diagram illustrating a sequence representation layer of a recurrent neural network, according to some implementations of the present specification;

[0019] FIG. 6 is a schematic flowchart illustrating another method for generating word vectors, according to some implementations of the present specification; and

[0020] FIG. 7 is a schematic structural diagram illustrating an apparatus for generating word vectors corresponding to FIG. 2, according to some implementations of the present specification.

DESCRIPTION OF IMPLEMENTATIONS

[0021] Implementations of the present specification provide methods, apparatuses, and devices for generating word vectors.

[0022] To enable a person skilled in the art to better understand technical solutions in the present specification, the following clearly and completely describes the technical solutions in the implementations of the present specification with reference to the drawings that accompany the implementations. Clearly, the described implementations are merely some rather than all of the implementations of the present application. Based on the implementations of the present specification, all other implementations obtained by a person of ordinary skill in the art without making innovative efforts shall fall within the protection scope of the present application.

[0023] FIG. 1 is a schematic diagram of an overall architecture involved in an actual application scenario of solutions of the present specification. The overall architecture relates primarily to a server that is used to train a recurrent neural network to generate word vectors. A feature vector of each word can be established based on an n-ary character, and a recurrent neural network can be trained based on the feature vectors and the context of the words. The feature vectors can be established by the server or another device.

[0024] The solutions of the present specification are applicable to languages that are formed by alphabetic letters, such as English, French, German, and Spanish, and are also applicable to languages that are formed by non-alphabetic elements but can be easily mapped to alphabetic letters, such as Chinese (which can be mapped to pinyin letters) and Japanese (which can be mapped to Roman letters). For ease of description, in the following implementations, the solutions of the present specification are mainly described with respect to scenarios of English.

[0025] FIG. 2 is a schematic flowchart illustrating a method for generating word vectors, according to some implementations of the present specification. From the perspective of a device, an actor that executes the process includes, for example, at least one of a personal computer, a large- or medium-sized computer, a computer cluster, a mobile phone, a tablet computer, a smart wearable device, a vehicle, etc.

[0026] The process in FIG. 2 can include the following steps:

[0027] S202: Obtain individual words by segmenting a corpus.

[0028] In some implementations of the present specification, more specifically, the words obtained can be at least some of the words that appear at least once in the corpus. To facilitate subsequent processing, each word can be stored in a vocabulary, and individual words can be read from the vocabulary for use.

[0029] It is worthwhile to note that, considering that if a word appears too few times in the corpus, the number of corresponding iterations in the subsequent processing is also small, and the training result reliability is relatively low, such a word can be screened out. In this case, more specifically, the obtained words are some words out of all the words that appear at least once in the corpus.

[0030] S204: Establish a feature vector for each obtained word based on respective n-ary character(s) corresponding to each word, each n-ary character representing n consecutive characters of a word corresponding to the n-ary character.

[0031] In this implementation of the present specification, characters of a word can include the characters constituting the word, or other characters to which the characters constituting the word are mapped. For example, the word “boy” includes characters “b”, “o”, and “y”.

[0032] To express the order of words, some mark character(s) can be added to the original word(s) based on certain rules, and these mark character(s) can also be considered characters of the word(s). For example, a mark character can be added to a

position such as a start position and/or an end position of the original word, and after such marking, the word “boy” can take the form of “#boy#”, and the two “#” can be considered characters of the word “boy”.

[0033] Further, n is an integer not less than 1. Use “#boy#” as an example, the word includes the following 3-ary characters: “#bo” (the 1st to 3rd characters), “boy” (the 2nd to 4th characters), and “oy#” (the 3rd to 5th characters); and includes the following 4-ary characters: “#boy” (the 1st to 4th characters) and “boy#” (the 2nd to 5th characters).

[0034] In some implementations of the present specification, the value of n can be dynamically adjusted. For the same word, at the time of determining the n-ary character corresponding to the word, n can have a single value (e.g., only 3-ary characters corresponding to the word are determined) or multiple values (e.g., both 3-ary characters and 4-ary characters corresponding to the word are determined).

[0035] To facilitate computer processing, n-ary characters can be represented based on specified codes (e.g., numbers). For example, different characters or different n-ary characters can be represented by different codes or code strings.

[0036] In some implementations of the present specification, the feature vector of a word can have values assigned to different dimensions to indicate the n-ary characters corresponding to the word. More precisely, the feature vector of the word can also be used to indicate the order of the n-ary characters corresponding to the word.

[0037] S206: Train a recurrent neural network based on the feature vectors of the words and of their context words in the corpus.

[0038] In some implementations of the present specification, the sequence representation layer of the recurrent neural network is used to process the sequence to obtain the global information of the sequence, and the content of the global information is affected by the order of the elements in the sequence. Specifically, in the context of the present specification, a sequence includes feature vectors of each context word associated with a current word (each individual word as obtained can be used as a current word) as elements, and global information can refer to overall semantics of all context words associated with the current word.

[0039] S208: Generate a word vector for each obtained word based on the feature vector of the word and the trained recurrent neural network.

[0040] By training the recurrent neural network, appropriate parameters can be

determined for the recurrent neural network, so that the recurrent neural network can more accurately depict the overall semantics of the context words and the corresponding semantics of the current words. The parameters include, for example, weight parameters and offset parameters.

[0041] The word vector can be obtained by making inference(s) from the feature vector based on the trained recurrent neural network.

[0042] By using the method in FIG. 2, the recurrent neural network can depict the overall semantic information of the context of the word through recurrent calculation, extract more semantic information of the context, and the n-ary character(s) can express the word more finely, so that the word vector can be more accurately generated.

[0043] Based on the method shown in FIG. 2, some implementations of the present specification further provides some implementations solutions of the method, as well as extension solutions, which are described below.

[0044] In some implementations of the present specification, for step S204, the establishing of a feature vector for each word based on the n-ary character(s) corresponding to the word can specifically include:

[0045] determining a total number of distinct n-ary characters in a collection of respective n-ary characters corresponding to each of the obtained words; and establishing, for each word, a feature vector whose dimensionality is determined based on the total number, where the feature vector can be assigned values to different dimensions to indicate the n-ary characters corresponding to the word.

[0046] For example, all of the n-ary characters corresponding to each and every obtained word are indexed starting from 0 at an increment of 1. The same n-ary characters have the same index number. Assuming that the total number is N_c, then the index number of the last n-ary character is $N_c - 1$. A feature vector whose dimension is N_c is established for each word. Specifically, assuming that n = 3, and that the indexes of all the 3-ary characters corresponding to a word are 2, 34, and 127, then the 2nd, the 34th , and the 127th elements in the feature vector established for that word can be 1, and the remaining elements are 0.

[0047] More intuitively, based on the above example, this implementation of the present specification provides a schematic diagram illustrating a feature vector of an English word in an application scenario, as shown in FIG. 3. The English word is

"boy", and a mark character "#" is added at the start position and the end position. f indicates the process of establishing a feature vector based on the word. The feature vector is, for example, a column vector that is established based on each 3-ary character of "boy". It can be seen that, the values of three elements in the feature vector are 1s, which respectively indicate the 3-ary characters "#bo", "boy", and "oy#"; and the values of other elements are 0s, which indicates that "boy" does not correspond to any other 3-ary character.

[0048] In this implementation of the present specification, when the recurrent neural network is trained, the goal is to ensure that the similarity between the feature vectors of the current word and its context word(s) to be relatively high after inferences made based on the trained recurrent neural network.

[0049] Further, the context word(s) is regarded as positive sample word(s), and, as a contrast, one or more negative sample words associated with the current word can be selected based on certain rule(s) to be involved in the training, so as to ensure fast training convergence and obtain more accurate training results. In this case, the goal can further include ensuring that the similarity between the feature vectors of the current word and the negative sample word(s) to be relatively low after inferences made based on the trained recurrent neural network. Negative sample words can be selected randomly in the corpus, or selected in the non-context words, etc. The present specification does not limit the specific ways of calculating such similarity. For example, the similarity can be calculated based on a cosine operation of the angle between vectors, the similarity can be calculated based on the square sum operation of the vectors, etc.

[0050] According to the above analysis, for step S206, the recurrent neural network is trained based on the feature vectors of the obtained words as well as the feature vectors of their context words in the corpus. Specifically, the training can include:

[0051] training the recurrent neural network based on the feature vectors of the obtained words as well as the feature vectors of their context words and their negative sample words in the corpus.

[0052] In some implementations of the present specification, the training process of the recurrent neural network can be iteratively performed. A relatively simple way is to traverse the corpus after word segmentation, and each time one of the obtained words is accessed, an iteration is performed, until the traversal is complete. The

recurrent neural network can then be considered to have been trained using the corpus.

[0053] Specifically, the training of the recurrent neural network based on the feature vectors of the obtained words and the feature vectors of their context words and negative sample words in the corpus can include:

[0054] traversing the corpus after word segmentation, and executing the following for a current word accessed during the traversal (the content of this execution corresponds to a process during one iteration):

[0055] for a current word obtained after word segmentation, determining one or more context words and negative sample words in the corpus; inputting a sequence formed by feature vector(s) of the context word(s) of the current word into a sequence representation layer of a recurrent neural network for recurrent calculation to obtain a first vector; inputting a feature vector of the current word into the full connection layer of the recurrent neural network for calculation to obtain a second vector, and inputting a feature vector of a negative sample word associated with the current word into the full connection layer of the recurrent neural network for calculation to obtain a third vector; and updating parameters of the recurrent neural network based on the first vector, the second vector, the third vector, and a specified loss function.

[0056] More intuitively, the above process is described with reference to FIG. 4. FIG. 4 is a schematic flowchart illustrating a principle of a method for generating word vectors based on a recurrent neural network in an actual application scenario, according to some implementations of the present specification.

[0057] The recurrent neural network in FIG. 4 mainly includes a sequence representation layer, a full connection layer, and a Softmax layer. In the process of training the recurrent neural network, the feature vector of the context word(s) is processed by the sequence presentation layer to extract the semantic information of the context word(s) as a whole, and the feature vector of the current word and its negative sample word(s) can be processed by the full connection layer. Detailed description is provided below.

[0058] In this implementation of the present specification, it is assumed that a sliding window is used to determine a context word, the center of the sliding window is the current word being accessed during the traversal, and the words in the sliding window other than the current word are context words. The feature vectors of all the

context words (the feature vector of each context word is one element in the sequence as described above) are sequentially inputted to the sequence representation layer, and then recurrent calculation can be performed according to the following formula:

$$s_t = \sigma\left(Ux_t + Ws_{t-1}\right)$$

$$o_t = softmax\left(Vs_t\right)$$

where x_t indicates the input unit of the sequence representation layer at the time t, i.e., the feature vector of the $t+1^{\text{th}}$ context word; s_t indicates the hidden unit of the sequence representation layer at the time t; o_t indicates a vector obtained through recurrent calculation of the feature vectors of the first $t+1$ context words; U, W, and V indicate weight parameters of the sequence representation layer; σ indicates an activation function, such as a tanh function or a ReLu function. The subscript of the parameter in the formula can start from 0.

[0059] To facilitate understanding of the formula used for recurrent calculation, some implementations of the present specification further provides a schematic structural diagram illustrating a sequence representation layer of a recurrent neural network, as shown in FIG. 5.

[0060] On the left side of FIG. 5, x indicates an input unit of a sequence representation layer; s indicates a hidden unit of the sequence representation layer; o indicates an output unit of the sequence representation layer; the hidden unit processes input data by using an activation function, and the output unit processes input data by using the $softmax$ function. The data calculated and output by the hidden unit at a previous time is fed back to the input of the hidden unit at a next time with a certain weight, so that the content of the entire sequence is reflected in the final output data of the sequence representation layer; U indicates a weight parameter from the input unit to the hidden unit; W indicates a weight parameter for feeding back from the hidden unit to the hidden unit at the next time; and V indicates a weight parameter from the hidden unit to the output unit.

[0061] On the right side of FIG. 5, the structure on the left side is expanded to illustrate the structure of the sequence input layer at 3 consecutive points of time and the principle for processing the sequential input of 3 elements. It can be seen that, at the $t-1$ time, the data that is output following computation by the hidden unit is denoted as s_{t-1}; at the t time, the feature vector x_t of the t^{th} context word

associated with the current word is the input, and the data that is output following computation by the hidden unit is denoted as $s_t = \sigma(Ux_t + Ws_{t-1})$, and the data that is output by the output unit is denoted as $o_t = softmax(Vs_t)$; at the t+1 time, the feature vector x_{t+1} of the t+1 th context word associated with the current word is the input, and the data that is output following computation by the hidden unit is denoted as $s_{t+1} = \sigma(Ux_{t+1} + Ws_t)$, and the data that is output by the output unit is denoted as $o_{t+1} = softmax(Vs_{t+1})$. By analogy and extension, after the feature vector of the last context word associated with the current word is input, a corresponding first vector can be output.

[0062] FIG. 4 also exemplarily illustrates the current word "liquid" in a corpus, 6 context words "as", "the", "vegan", "gelatin", "substitute", and "absorbs" of the current word in the corpus, and 2 negative sample words "year" and "make" of the current word in the corpus. FIG. 4 shows the output unit corresponding to the last context word "absorbs" at the sequence representation layer, where the output unit outputs a first vector corresponding to the current word "liquid".

[0063] The feature vector of the current word can be input to the full connection layer, and can be calculated according to, for example, the following formula:

$$w = \sigma(\varsigma \cdot q + \tau)$$

where w indicates the second vector that is output by the full connection layer after processing the feature vector of the current word; ς indicates the weight parameter of the full connection layer; q indicates the feature vector of the current word; and τ indicates the offset parameter of the full connection layer.

[0064] Similarly, for each negative sample word, a respective feature vector can be input to the full connection layer and processed in a way as the current word is processed, to obtain a third vector, and the third vector corresponding to the m th negative sample word is denoted as $w_m^{'}$.

[0065] Further, the updating of parameters of the recurrent neural network based on the first vector, the second vector, the third vector(s), and a specified loss function can include, for example, calculating a first similarity between the second vector and the first vector, and a second similarity between the third vector and the first vector; and updating parameters of the recurrent neural network based on the first similarity, the

second similarity, and the specified loss function.

[0066] A loss function is used as an example. The loss function can be, for example:

$$l(w,c;U,W,V,\varsigma,\tau)=log\left(1+\sum_{m=1}^{\lambda}exp\left(-\gamma\cdot\left(s(w,c)-s\left(w_m^{'},c\right)\right)\right)\right)$$

where c indicates the first vector; w indicates the second vector; $w_m^{'}$ indicates the third vector corresponding to the m^{th} negative sample word; U, W, and V indicate weight parameters of the sequence representation layer; ς indicates a weight parameter of the full connection layer; τ indicates an offset parameter of the full connection layer; γ indicates a hyperparameter; s indicates a similarity calculation function; and λ indicates the number of negative sample words.

[0067] In practice, if no negative sample word is used, the term for calculating the similarity between the first vector and the third vector(s) can be removed from the loss function used.

[0068] In some implementations of the present specification, after training of the recurrent neural network, a word vector can be generated by making inference(s) from the feature vector. Specifically, for step S208, the generating of a word vector for each word based on the feature vector of each word and the trained recurrent neural network can specifically include:

[0069] inputting the feature vector of each obtained word into the full connection layer of the trained recurrent neural network for calculation to obtain a vector output, which is considered a corresponding word vector.

[0070] Based on the same idea, some implementations of the present specification provide another method for generating word vectors, which is an example implementation of the method for generating word vectors in FIG. 2. FIG. 6 is a schematic flowchart illustrating such a method for generating word vectors.

[0071] The process in FIG. 6 includes following steps:

Step 1: Establishing a vocabulary of words obtained by segmenting a corpus, where the words do not include a word that occurs less than a predetermined number of times in the corpus; go to step 2;

Step 2: Determining a total number of n-ary characters corresponding to each of the obtained words, where the same n-ary characters are counted only once, and each n-ary character represents n consecutive characters of a word corresponding to the

n-ary character; go to step 3;

Step 3: Establishing, based on the n-ary character(s) corresponding to each word, a feature vector whose dimension is the total number described above, where each dimension of the feature vector corresponds to a different n-ary character, and the value of each dimension indicates whether its corresponding n-ary character is mapped to the word corresponding to the feature vector.

Step 4: Traversing the corpus on which word segmentation is performed, and performing step 5 on the current word being accessed during the traversal, and if the traversal is completed, performing step 6; otherwise continuing the traversal.

Step 5: Taking the current word as a center, sliding towards both sides for at most k words to establish a window, and using the words in the window except the current word as context words, and inputting a sequence formed by feature vectors of all the context words into a sequence representation layer of a recurrent neural network for recurrent calculation to obtain a first vector; inputting the feature vector of the current word and the feature vector of a negative sample word selected in the corpus into a full connection layer of the recurrent neural network for calculation to obtain a second vector and a third vector; and updating parameters of the recurrent neural network based on the first vector, the second vector, the third vector, and a specified loss function.

The recurrent calculation is performed according to the following formula:

$$s_t = \sigma\left(Ux_t + Ws_{t-1}\right)$$

$$o_t = softmax\left(Vs_t\right)$$

The loss function includes:

$$l\left(w,c;U,W,V,\varsigma,\tau\right) = log\left(1+\sum_{m=1}^{\lambda} exp\left(-\gamma\cdot\left(s\left(w,c\right)-s\left(w_m^{'},c\right)\right)\right)\right)$$

where x_t indicates an input unit of the sequence representation layer at time t, that is, the feature vector of the t+1$^{\text{th}}$ context word associated with the current word; s_t indicates a hidden unit of the sequence representation layer at the time t; o_t indicates a vector obtained through recurrent calculation based on the feature vectors of the first t+1 context words associated with the current word; U, W, and V indicate weight parameters of the sequence representation layer; σ indicates an

activation function; c indicates the first vector; w indicates the second vector; $w_m^{'}$ indicates the third vector corresponding to the m^{th} negative sample word; ς indicates a weight parameter of the full connection layer; τ indicates an offset parameter of the full connection layer; γ indicates a hyperparameter; s indicates a similarity calculation function; and λ indicates the number of negative sample words.

Step 6: Inputting the feature vector of each obtained word into the full connection layer of the trained recurrent neural network for calculation to obtain corresponding word vectors.

[0072] The steps in this method for generating word vectors can be performed by same or different modules, which are not specifically limited in the present specification.

[0073] The foregoing has described the method for generating word vectors according to some implementations of the present specification. Based on the same idea, some implementations of the present specification further provide a corresponding apparatus, as shown in FIG. 7.

[0074] FIG. 7 is a schematic structural diagram illustrating an apparatus for generating word vectors corresponding to FIG. 2, according to some implementations of the present specification. The apparatus can be located in the entity to execute the process in FIG. 2 and includes:

an acquisition module 701, configured to obtain individual words by segmenting a corpus;

an establishment module 702, configured to establish a feature vector for each word based on n-ary character(s) corresponding to each word, each n-ary character representing n consecutive characters of the word corresponding to the n-ary character;

a training module 703, configured to train a recurrent neural network based on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus; and

a generation module 704, configured to generate a word vector for each word based on the feature vector of each word and the trained recurrent neural network.

[0075] Optionally, characters of a word include each character constituting the word and mark characters added to a start position and/or an end position of the word.

[0076] Optionally, the establishment module 702's establishing of a feature vector for each obtained word based on n-ary characters corresponding to the obtained word specifically includes:

determining, by the establishment module 702, a total number of distinct n-ary characters in a collection of respective n-ary characters corresponding to each of the obtained words; and

establishing, for each obtained word, a feature vector whose dimensionality is determined based on the total number, where the feature vector can be assigned values to its different dimensions to indicate n-ary characters corresponding to the obtained word.

[0077] Optionally, the training module 703's training of the recurrent neural network based on the feature vectors of the words and the feature vectors of the context words of the words in the corpus specifically includes:

[0078] training, by the training module 703, the recurrent neural network based on the feature vectors of the words and the feature vectors of the context words and the negative sample words of the words in the corpus.

[0079] Optionally, the training module 703's training of the recurrent neural network based on the feature vectors of the words and the feature vectors of the context words and the negative sample words of the words in the corpus specifically includes:

[0080] traversing, by the training module 703, the corpus after word segmentation, and executing the following when a current word is accessed during the traversal:

determining one or more context words and negative sample words associated with the current word in the corpus after word segmentation;

inputting a sequence formed by feature vectors of the context words associated with the current word into a sequence representation layer of the recurrent neural network for recurrent calculation to obtain a first vector;

inputting a feature vector of the current word into the full connection layer of the recurrent neural network for calculation to obtain a second vector, and inputting a feature vector of a negative sample word associated with the current word into the full connection layer of the recurrent neural network for calculation to obtain a third vector; and

updating parameters of the recurrent neural network based on the first vector, the

second vector, the third vector, and a specified loss function.

[0081] Optionally, the training module 703's performing of recurrent calculation specifically includes:

[0082] performing, by the training module 703, recurrent calculation according to the following formula:

$$s_t = \sigma\left(Ux_t + Ws_{t-1}\right)$$

$$o_t = softmax\left(Vs_t\right)$$

where x_t indicates an input unit of the sequence representation layer at the time t, that is, a feature vector of the $t+1^{\text{th}}$ context word associated with the current word; s_t indicates a hidden unit of the sequence representation layer at the time t; o_t indicates a vector obtained through recurrent calculation of the feature vectors of the first $t+1$ context words associated with the current word; U, W, and V indicate weight parameters of the sequence representation layer; and σ indicates an activation function.

[0083] Optionally, the training module 703's updating of parameters of the recurrent neural network based on the first vector, the second vector, the third vector, and a specified loss function specifically includes:

calculating, by the training unit 703, a first similarity between the second vector and the first vector, and a second similarity between the third vector and the first vector; and

updating parameters of the recurrent neural network based on the first similarity, the second similarity, and the specified loss function.

[0084] Optionally, the loss function specifically includes:

$$l\left(w,c;U,W,V,\varsigma,\tau\right) = log\left(1+\sum_{m=1}^{\lambda} exp\left(-\gamma\cdot\left(s\left(w,c\right)-s\left(w_m^{'},c\right)\right)\right)\right)$$

where c indicates the first vector; w indicates the second vector; $w_m^{'}$ indicates the third vector corresponding to the m^{th} negative sample word; U, W, and V indicate weight parameters of the sequence representation layer; ς indicates a weight parameter of the full connection layer; τ indicates an offset parameter of the full connection layer; γ indicates a hyperparameter; s indicates a similarity calculation function; and λ indicates the number of negative sample words.

[0085] Optionally, the generation module 704's generating of a word vector for each obtained word based on the feature vector of the obtained word and the trained recurrent neural network specifically includes:

[0086] inputting, by the generation module 704, the feature vector of each obtained word into the full connection layer of the trained recurrent neural network for calculation to obtain a vector output after calculation as a corresponding word vector.

[0087] Based on the same idea, some implementations of the present specification further provide a device for generating word vectors, including: at least one processor, and a memory communicatively connected to the at least one processor, where the memory stores instructions that can be executed by the at least one processor, and the instructions are executed by the at least one processor to enable the at least one processor to: obtain individual words by segmenting a corpus; establish a feature vector for each obtained word based on n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word corresponding to the n-ary character; train a recurrent neural network based on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus; and generate a word vector for each obtained word based on the feature vector of the obtained word and the trained recurrent neural network.

[0088] Based on the same idea, some implementations of the present specification further provide a non-volatile computer storage medium, where the storage medium stores computer executable instructions that are used to: obtain individual words by segmenting a corpus; establish a feature vector for each obtained word based on n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word corresponding to the n-ary character; train a recurrent neural network based on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus; and generate a word vector for each obtained word based on the feature vector of the obtained word and the trained recurrent neural network.

[0089] Specific implementations of the present specification are described above. Other implementations fall within the scope of the appended claims. In some situations, the actions or steps described in the claims can be performed in an order

CLAIMS

What is claimed is:

1. A method for generating word vectors, comprising:

obtaining individual words by segmenting a corpus;

establishing a feature vector for each obtained word based, at least in part, on one or more n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word;

training a recurrent neural network based, at least in part, on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus by:

traversing the corpus after word segmentation, and performing the following actions in response to that a current word is accessed during the traversal:

determining one or more context words and negative sample words associated with the current word in the corpus;

inputting a sequence formed by feature vectors of the context words associated with the current word into a sequence representation layer of the recurrent neural network to obtain a first vector;

inputting a feature vector of the current word into a full connection layer of the recurrent neural network to obtain a second vector; and

inputting a feature vector of a negative sample word associated with the current word into the full connection layer of the recurrent neural network to obtain a third vector; and

updating parameters of the recurrent neural network based, at least in part, on the first vector, the second vector, the third vector, and a loss function; and

generating a word vector for each obtained word based, at least in part, on the feature vector of the obtained word and the trained recurrent neural network.

2. The method according to claim 1, wherein characters of an obtained word comprise each character constituting the obtained word and mark characters added to a start position and/or an end position of the obtained word.

3. The method according to claim 1, wherein establishing the feature vector for each obtained word based, at least in part, on one or more n-ary characters corresponding to the obtained word comprises:

determining a total number of distinct n-ary characters in a collection of respective n-ary characters corresponding to each of the obtained words; and

establishing, for each obtained word, a feature vector whose dimensionality is determined based, at least in part, on the total number.

4. The method according to claim 3, wherein the number of dimensions of the feature vector is the total number.

5. The method according to claim 3, wherein values are assigned to individual dimensions of the feature vector to indicate a mapping between n-ary characters and the obtained word.

6. The method according to claim 1, further comprising performing recurrent calculation using the sequence representation layer of the recurrent neural network based on at least a hidden unit and an activation function.

7. The method according to claim 1, wherein updating the parameters of the recurrent neural network based on the first vector, the second vector, the third vector, and the loss function comprises:

calculating a first similarity between the second vector and the first vector, and a second similarity between the third vector and the first vector; and

updating parameters of the recurrent neural network based, at least in part, on the first similarity, the second similarity, and the loss function.

8. The method according to claim 1, wherein the loss function includes at least a weight parameter of the sequence layer, a weight parameter of the full connection layer, an offset parameter of the full connection layer, a hyperparameter, and a similarity calculation function.

9. The method according to claim 1, wherein generating a word vector for each obtained word based, at least in part, on the feature vector of the obtained word and the trained recurrent neural network specifically comprises:

inputting the feature vector of each obtained word into the full connection layer of the trained recurrent neural network to obtain a vector output.

10. An apparatus for generating word vectors, comprising:

an acquisition module, configured to obtain individual words by segmenting a corpus;

an establishment module, configured to establish a feature vector for each obtained word based, at least in part, on one or more n-ary characters corresponding to the obtained word, each n-ary character representing n consecutive characters of a word;

a training module, configured to train a recurrent neural network based, at least in part, on the feature vectors of the obtained words and the feature vectors of context words associated with each obtained word in the corpus by:

traversing the corpus after word segmentation, and performing the following actions in response to a current word is accessed during the traversal:

determining one or more context words and negative sample words associated with the current word in the corpus;

inputting a sequence formed by feature vectors of the context words associated with the current word into a sequence representation layer of the recurrent neural network to obtain a first vector;

inputting a feature vector of the current word into a full connection layer of the recurrent neural network to obtain a second vector; and

inputting a feature vector of a negative sample word associated with the current word into the full connection layer of the recurrent neural network to obtain a third vector; and

updating parameters of the recurrent neural network based, at least in part, on the first vector, the second vector, the third vector, and a loss function; and

a generation module, configured to generate a word vector for each obtained word based, at least in part, on the feature vector of the obtained word and the trained recurrent neural network.

11. The apparatus according to claim 10, wherein characters of an obtained word comprise each character constituting the obtained word and mark characters added to a start position and/or an end position of the obtained word.

12. The apparatus according to claim 10, wherein the establishment module is further configured to establish a feature vector for each obtained word based, at least in part, on n-ary characters corresponding to the obtained word by:

determining, by the establishment module, a total number of distinct n-ary characters in a collection of respective n-ary characters corresponding to each of the obtained words; and

establishing, for each obtained word, a feature vector whose dimensionality is determined based, at least in part, on the total number.

13. The apparatus according to claim 12, wherein the number of dimensions of the feature vector is the total number.

14. The apparatus according to claim 13, wherein values are assigned to individual dimensions of the feature vector to indicate a mapping between n-ary characters and the obtained word.

15. The apparatus according to claim 10, wherein the training module is further configured to perform one or more recurrent calculations according to the following formula:

$$s_t = \sigma\left(Ux_t + Ws_{t-1}\right)$$

$$o_t = softmax\left(Vs_t\right)$$

wherein x_t indicates an input unit of the sequence representation layer at the time t, that is, a feature vector of the $t+1^{\text{th}}$ context word associated with the current word; s_t indicates a hidden unit of the sequence representation layer at the time t; o_t indicates a vector obtained through recurrent calculation of the feature vectors of the first $t+1$ context words associated with the current word; U, W, and V indicate weight parameters of the sequence representation layer; and σ indicates an activation function.

16. The apparatus according to claim 10, wherein updating the parameters of the recurrent neural network based, at least in part, on the first vector, the second vector, the third vector, and the loss function comprises:

calculating, by the training unit, a first similarity between the second vector and the first vector, and a second similarity between the third vector and the first vector; and

updating the parameters of the recurrent neural network based, at least in part, on the first similarity, the second similarity, and the loss function.

17. The apparatus according to claim 10, wherein the loss function comprises:

$$l\left(w,c;U,W,V,\varsigma,\tau\right) = log\left(1+\sum_{m=1}^{\lambda} exp\left(-\gamma\cdot\left(s\left(w,c\right)-s\left(w_m^{'},c\right)\right)\right)\right)$$

wherein c indicates the first vector; w indicates the second vector; $w_m^{'}$ indicates the third vector corresponding to the m^{th} negative sample word; U, W, and V indicate weight parameters of the sequence representation layer; ς indicates a weight parameter of the full connection layer; τ indicates an offset parameter of the full

connection layer; γ indicates a hyperparameter; s indicates a similarity calculation function; and λ indicates the number of negative sample words.

18. The apparatus according to claim 10, wherein the generation module is further configured to generate a word vector for each obtained word based, at least in part, on the feature vector of the obtained word and the trained recurrent neural network by:

inputting, by the generation module, the feature vector of each obtained word into the full connection layer of the trained recurrent neural network to obtain a vector output.

19. A method for generating word vectors, comprising:

Step 1: Establishing a vocabulary of words obtained by segmenting a corpus, wherein the obtained words exclude a word that appears for less than a predetermined number of times in the corpus;

Step 2: Determining a total number of n-ary characters corresponding to all of the obtained words, wherein same n-ary characters are counted once, and each n-ary character represents n consecutive characters of an obtained word corresponding to the n-ary character;

Step 3: Establishing, for each obtained word, based on each n-ary character corresponding to the obtained word, a feature vector whose dimensionality is the total number, wherein each dimension of the feature vector corresponds to a different n-ary character, and the value of each dimension indicates whether a corresponding n-ary character is mapped to the obtained word corresponding to the feature vector;

Step 4: Traversing the corpus on which word segmentation is performed, and performing step 5 with respect to a current word accessed during the traversal, and if the traversal is completed, performing step 6; otherwise continuing the traversal;

Step 5: Using the current word as a center, sliding towards both sides for at most k words to establish a window, and using the words in the window except the current word as context words, and inputting a sequence formed by feature vectors of

all the context words into a sequence representation layer of a recurrent neural network for recurrent calculation to obtain a first vector; inputting a feature vector of the current word and a feature vector of a negative sample word selected in the corpus into a full connection layer of the recurrent neural network for calculation to obtain a second vector and a third vector; and updating parameters of the recurrent neural network based on the first vector, the second vector, the third vector, and a specified loss function, wherein

the recurrent calculation is performed according to the following formula:

$$s_t = \sigma\left(Ux_t + Ws_{t-1}\right)$$

$$o_t = softmax\left(Vs_t\right)$$

and the loss function includes:

$$l\left(w, c; U, W, V, \varsigma, \tau\right) = log\left(1 + \sum_{m=1}^{\lambda} exp\left(-\gamma \cdot \left(s\left(w, c\right) - s\left(w_m^{'}, c\right)\right)\right)\right)$$

wherein x_t indicates an input unit of the sequence representation layer at the time t, that is, a feature vector of the $t+1$ th context word associated with the current word; s_t indicates a hidden unit of the sequence representation layer at the time t; o_t indicates a vector obtained through recurrent calculation based on the feature vectors of the first $t+1$ context words associated with the current word; U, W, and V indicate weight parameters of the sequence representation layer; σ indicates an activation function; c indicates the first vector; w indicates the second vector; $w_m^{'}$ indicates the third vector corresponding to the m th negative sample word; ς indicates a weight parameter of the full connection layer; τ indicates an offset parameter of the full connection layer; γ indicates a hyperparameter; s indicates a similarity calculation function; and λ indicates the number of negative sample words; and

Step 6: Inputting the feature vector of each obtained word into the full connection layer of the trained recurrent neural network for calculation to obtain corresponding word vectors.