BAB 2
TINJAUAN PUSTAKA

2.1 Named Entity Recognition

Named Entity Recognition atau *NER* merupakan salah satu tugas dari *natural language processing* [1] yang bertujuan dalam mengenali unit informasi seperti nama termasuk nama orang, organisasi dan nama lokasi, dan ekspresi numerik termasuk waktu, tanggal, dan ekspresi persen [1]. Jadi, *named entity recognition* bertujuan untuk mengenali entitas apapun yang memiliki sebuah informasi nama.

2.2 Berita

Berita (*news*) adalah laporan mengenai suatu peristiwa atau kejadian yang terbaru (aktual), laporan mengenai fakta -fakta yang aktual, menarik perhatian, dinilai penting, atau luar biasa [8]. Pada penelitian ini, berita yang digunakan dikhususkan pada kategori tertentu, yaitu berita politik. Hal tersebut dilakukan karena pada sistem NER (*Name Entity Recognition*) pada penelitian sebelumnya menggunakan berita dengan kategori politik.

2.3 Korpus

Korpus menurut Kamus Besar Bahasa Indonesia (KBBI) adalah kumpulan ujaran yang tertulis atau lisan yang digunakan untuk menyokong atau menguji hipotesis tentang struktur bahasa. Korpus juga bisa diartikan sebagai data yang dipakai sebagai sumber bahan penelitian.

Pada penelitian ini korpus yang digunakan adalah hasil dari penelitian Rusliani [2] dan Fachri [5].

2.4 Tokenisasi

Tokenisasi merupakan tahap pemisahan teks menjadi kata per kata dengan menggunakan sebuah pemisah. Menurut Amin, token seringkali disebut sebagai istilah (*term*) atau kata. Sebuah token merupakan suatu urutan karakter dari

dokumen tertentu yang dikelompokkan sebagai unit semantik yang berguna untuk diproses [9]. Dalam pemisahan kata tersebut dalam penelitian ini menggunakan sebuah *library nltk.word_tokenize*. Tahap yang dilakukan *library* tersebut adalah sebagai berikut.

1. Memisahkan token dengan space atau tab
2. karakter tanda baca dianggap sebagai token terpisah
3. kata dan tanda baca dianggap satu token jika diikuti dengan kata lainnya, mis : S.T, 2.3, dan lain sebagainya.

2.5 Long Short Term Memory

Recurrent neural networks (RNN) adalah sebuah metode yang beroperasi untuk data *sequence.* Metode ini mengambil input dari vektor *sequence* (x_1, x_2, ..., x_n) dan menjadi sequence yang lain (h_1, h_2, ..., h_n) [3]. metode RNN tidak bisa digunakan dalam long-term dependency yang menimbulkan masalah vanishing gradient [1]. Oleh sebab itu *long short term memory* (LSTM) dikembangkan untuk mengatasi permasalahan *vanishing gradient* [1]. LSTM mengganti hidden unit pada arsitektur RNN dengan unit yang bisa disebut *memory block* yang terdiri dari empat komponen yaitu : *input gate*, *output gate*, *forget gate*, dan *memory cell* [3]. Rumus dari ke-empat komponen tersebut adalah sebagai berikut.

$$i_t = \sigma\,(W_i x_t + U_i h_{t-1} + b_i)$$

$$f_t = \sigma\,(W_f x_t + U_f h_{t-1} + b_f)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh\,(W_c x_t + U_c h_{t-1} + b_c)$$

$$o_t = \sigma\,(W_o x_t + U_o h_{t-1} + b_o)$$

$$h_t = o_t \odot \tanh(c_t)$$

Dimana σ adalah fungsi sigmoid dengan persamaan

$$f(x) = \frac{1}{1 + \exp(-x)}$$

Dan tanh dengan persamaan

$$tanh(x) = \frac{2}{1 + \exp(-2x)} - 1$$

W dan U adalah bobot matriks; b adalah vektor bias; ⊙ element wise production; dan i, f, o, c adalah fungsi aktivasi *input gate*, *forget gate*, *output gate*, *cell state* pada setiap *timestep* (t).

Dalam melakukan prediksi LSTM menggunakan softmax function sebagai *output layer* dengan persamaan

$$softmax(x_j) = \frac{\exp(x_j)}{\sum_{k=1}^{K} \exp(x_k)} \; untuk \; j = 1, \ldots, K$$

Setelah di klasifikasi, hitung turunan dari *output layer* dengan menggunakan persamaan

$$dz_t = y_t - z_t$$
$$dW_{hz} = \sum_t \mathbf{h}_t dz_t$$
$$d\mathbf{h}_T = W_{hz} dz_T$$

Setelah itu, backpropagate LSTM dengan persamaan

$$d\mathbf{o}_t = tanh(\mathbf{c}_t) d\mathbf{h}_t$$
$$d\mathbf{c}_t = \left(1 - tanh(\mathbf{c}_t)^2\right) \mathbf{o}_t d\mathbf{h}_t$$
$$d\mathbf{f}_t = \mathbf{c}_{t-1} d\mathbf{c}_t$$
$$d\mathbf{c}_{t-1} += \mathbf{f}_t \circ d\mathbf{c}_t$$
$$d\mathbf{i}_t = \mathbf{g}_t d\mathbf{c}_t$$
$$d\mathbf{g}_t = \mathbf{i}_t d\mathbf{c}_t$$

Backpropagate bobot

$$dW_{xo} = \sum_t (1 - \mathbf{o}_t^2) \mathbf{x}_t d\mathbf{o}_t$$
$$dW_{xi} = \sum_t \mathbf{i}_t (1 - \mathbf{i}_t) \mathbf{x}_t d\mathbf{i}_t$$
$$dW_{xf} = \sum_t \mathbf{f}_t (1 - \mathbf{f}_t) \mathbf{x}_t d\mathbf{f}_t$$
$$dW_{xc} = \sum \mathbf{g}_t (1 - \mathbf{g}_t) \mathbf{x}_t d\mathbf{g}_t$$

$$dW_{ho} = \sum_t (1 - \mathbf{o}_t^2)\mathbf{h}_{t-1} d\mathbf{o}_t$$
$$dW_{hi} = \sum_t \mathbf{i}_t(1 - \mathbf{i}_t)\mathbf{h}_{t-1} d\mathbf{i}_t$$
$$dW_{hf} = \sum_t \mathbf{f}_t(1 - \mathbf{f}_t)\mathbf{h}_{t-1} d\mathbf{f}_t$$
$$dW_{hc} = \sum_t \mathbf{g}_t(1 - \mathbf{g}_t)\mathbf{h}_{t-1} d\mathbf{g}_t$$

Backpropagate hidden output layer

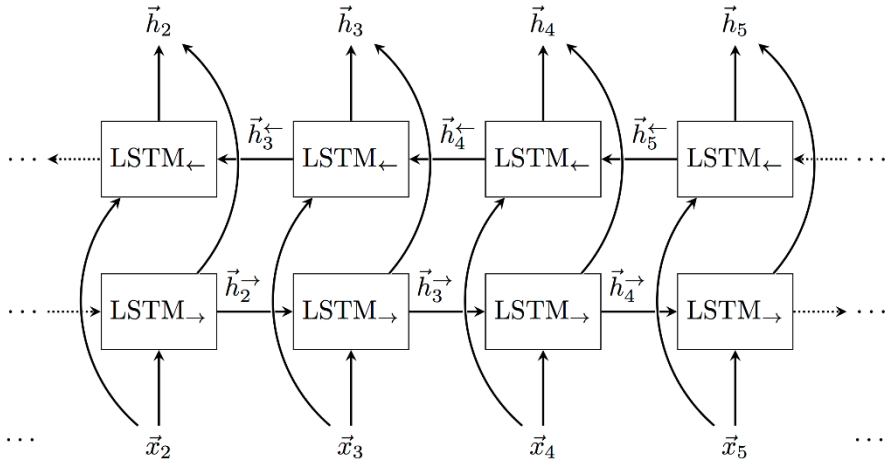
$$\begin{aligned} d\mathbf{h}_{t-1} &= (1 - \mathbf{o}_t^2)W_{ho} d\mathbf{o}_t + \mathbf{i}_t(1 - \mathbf{i}_t)W_{hi} d\mathbf{i}_t \\ &\quad + \mathbf{f}_t(1 - \mathbf{f}_t)W_{hf} d\mathbf{f}_t + \mathbf{g}_t(1 - \mathbf{g}_t)W_{hc} d\mathbf{g}_t \\ d\mathbf{h}_{t-1} &= d\mathbf{h}_{k-1} + W_{hz} dz_{t-1} \end{aligned}$$

Update paramater menggunakan persamaan berikut

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \eta d\boldsymbol{\theta}$$

2.6 *Bidirectional LSTM*

Bidirectional LSTM memanfaatkan konteks sebelumnya dan konteks setelahnya dengan memproses data dari dua arah dengan *hidden layer* terpisah [10] [1]. *Forward layer* untuk merepresentasikan konteks sebelumnya, dan *backward layer* untuk merepresentasikan konteks setelahnya [1]. Keluaran dari kombinasi dua arah *hidden layer* $\vec{h}_t$ dan $\overleftarrow{h_t}$ adalah : $y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h_t}$.



Gambar 2.1 Arsitektur *Bidirectional LSTM*

2.7 Conditional Random Field

Conditional Random Field merupakan sebuah model probabilistik yang digunakan untuk memprediksi data terstruktur yang telah digunakan dalam berbagai tugas, seperti *computer vision, natural language processing* [1].

Model CRF melakukan *training* untuk memprediksi sebuah vektor y {y0,y1,y2,..,yT} dari sebuah kalimat X {x0, x1, x2,..xT} dengan

$$p(y|x) = \frac{e^{score(x,y)}}{\sum_{y'} e^{score(x,y')}}$$

Dimana untuk mencari $score(x, y)$ dapat menggunakan

$$score(x, y) = \sum_{i=0}^{T} A_{yi,yi+1} + \sum_{i=1}^{T} P_{i,yi}$$

Dimana $A_{yi,yi+1}$ adalah probabilitas emisi yang mewakili skor transisi dari tag *i* ke tag *j*. $P_{i,yi}$ adalah probabilitas transisi yang mewakili skor transisi dari tag *j* ke kata *i*.

2.8 Bidirectional LSTM-CRF

Bidirectional LSTM-CRF adalah sebuah kombinasi metode antara *bidirectional LSTM* dan model *CRF*. Vektor kata di hitung dengan *bidirectional LSTM* untuk menghasilkan skor yang mewakili kemungkinan tag pada setiap kata didalam kalimat. Artinya $P_{i,yi}$ diganti dengan hasil dari *Bidirectional LSTM* [3]. Sehingga pada model CRF hanya menghitung $A_{yi,yi+1}$.

2.9 Penelitian Terdahulu

Beberapa penelitian terkait *Name Entity Recognition* (NER) sudah dilakukan oleh beberapa peneliti terdahulu. Berikut penelitian terdahulu yang sudah dilakukan :

1. Pada penelitian yang dilakukan oleh Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, dan Chris Dyer dengan judul "Neural Architectures for Named Entity Recognition" pada tahun 2016 mengenalkan dua neural arsitektur baru yaitu berbasis pada bidirectional lstm

dan conditional random field, dan pendekatan berbasis transisi dalam membangun dan memberikan label segmen yang terinspirasi oleh *shift-reduce-parsers*. Akurasi yang diberikan oleh *bidirectional lstm-crf* adalah 90.94% [3].

2. Pada penelitian yang dilakukan oleh Anh L. T., Arkhipov M.Y., dan Burtsev M. S. dengan judul "Application of a Hybrid Bi-LSTM-CRF model to the task of Russian Named Entity Recognition" pada tahun 2017 menggunakan metode *bidirectional lstm-crf* dan penambahan *word embedding* eksternal. Model tersebut dievaluasi dengan tiga dataset yaitu : Gareev's dataset (87.17%), Person-1000 (99.26%) dan FactRuEval-2016 (82.10%) [1].

2.10 Pemodelan

Pemodelan adalah sebuah metode yang digunakan dalam memodelkan pada aplikasi NER yang akan dibangun. Berikut adalah pemodelan yang digunakan dalam penelitian yang dibangun.

2.10.1 Diagram UML

Diagram *Unifed Modelling Language* (UML) adalah sekumpulan alat yang digunakan untuk melakukan abstraksi terhadap sebuah sistem atau perangkat lunak berbasis objek [11] [12]. Dalam UML sendiri terdapat beberapa diagram seperti *class diagram*, *use case diagram*, *activity diagram* dan *sequence diagram*.

2.10.1.1 *Class Diagram*

Class diagram atau diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem [11] [12]. Class diagram memiliki tiga area pokok, berikut dibawah ini menunjukan area pokok tersebut.

1. *Class name* atau nama kelas adalah kumpulan dari objek-objek dengan karakteristik (atribut dan operasi) yang sama [11] [12].
2. *Attributes* adalah variabel global yang dimiliki sebuah kelas.

3. *Operations* atau metode adalah fungsi atau prosedur yang dimiliki sebuah kelas.

Pada penelitian ini class diagram digunakan untuk menggambarkan interaksi antar kelas di dalam sistem.

2.10.1.2 Use Case Diagram

Use case diagram merupakan pemodelan untuk menggambarkan aktifitas (*behavior*) sistem yang akan dibuat [11] [12]. Use case diagram memiliki beberapa komponen, berikut dibawah ini menunjukan komponen-komponen tersebut.

1. *Actor* adalah pihak atau pengguna yang mengakses atau yang berinteraksi dengan use case.
2. *Use case* adalah gambaran apa saja yang bisa dilakukan oleh sistem atau gambaran fungsional sistem.
3. *Association* adalah sebuah komponen yang berfungsi untuk menghubungkan antara *actor* dengan *use case.*
4. *System boundary* adalah gambaran batasan sistem terhadap lingkungannya.

Pada penelitian ini *use case diagram* digunakan untuk menggambarkan fungsional sistem dan pengguna yang mengakses fungsional dari sistem.

2.10.1.3 Activity Diagram

Activity diagram atau diagram aktivitas adalah sebuah diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis [11] [12]. Diagram ini digunakan untuk melihat bagaimana sistem bekerja ketika dieksekusi [11] [12].

Pada penelitian ini *activity diagram* digunakan untuk menggambarkan aktivitas dari sistem.

2.10.1.4 Sequence Diagram

Sequence diagram atau diagram sekuen adalah diagram yang menggambarkan perilaku objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek [11] [12].

Sequence diagram memiliki beberapa komponen, berikut dibawah ini menunjukan komponen-komponen tersebut.

1. *Actor* adalah komponen yang mewakili pihak atau pengguna yang berinteraksi dengan sistem. Actor juga bisa disebut *object* dari luar sistem.
2. *Object* adalah komponen yang mewakili sebuah kelas yang mendemonstrasikan bagaimana sebuah kelas berperilaku pada sebuah sistem.
3. *Activation boxes* adalah komponen yang menggambarkan waktu aktif atau keadaan aktif suatu *object* dan berinteraksi melalui *messages*.
4. *Lifeline* adalah komponen yang menyatakan kehidupan suatu *object* dalam berinteraksi.
5. *Messages* adalah komponen yang menyatakan proses berinteraksi antar *object.*

Pada penelitian ini *sequence diagram* digunakan untuk menggambarkan interaksi antara aktor dengan objek dan objek dengan objek.

2.11 Bahasa Pemrograman

Program adalah algoritma yang ditulis dalam bahasa komputer. Pemrograman adalah kegiatan merancang dan menulis program. Program ditulis dan dirancang menggunakan dengan suatu bahasa pemrogramaman. Bahasa pemrograman adalah bahasa komputer yang digunakan untuk menulis sekumpulan perintah atau intruksi yang bisa dimengerti oleh komputer.

2.12 Python

Python adalah bahasa pemrograman interpretatif multiguna . Python pertama kali muncul pada tahun 1991, dirancang oleh Guido van Rossum [13]. Bahasa python mendukung hampir semua sistem operasi. Python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks. Contoh kode pada bahasa python adalah sebagai berikut.

print (“hello world”)
hello world

2.13 Perangkat Lunak Pembangunan

Perangkat lunak adalah suatu perintah program dalam sebuah komputer, yang jika dieksekusi oleh penggunanya dapat memberikan fungsi yang diinginkan oleh penggunanya. Dengan demikian perangkat lunak ini berfungsi untuk memberi perintah kepada komputer sesuai dengan keinginan pengguna. Dalam pembangunan aplikasi NER, peneliti menggunakan Spyder sebagai penyunting kode bahasa program.

2.13.1 Spyder

Spyder adalah sebuah IDE yang digunakan dalam membangun aplikasi scientific dengan pengeditan lanjutan, pengujian interaktif, debugging dan fitur introspeksi kesalahan dalam penyuntingan kode. Spyder juga digunakan dalam komputasi numerik berkat dukungan dari IPython dan memiliki pustaka python yang populer seperti NumPy, SciPy, dan matplotlib.

2.13.2 Flask

Flask adalah sebuah aplikasi microframework untuk bahasa Python yang dibuat dengan toolkit *Werkzeug* dan template *Jinja2* [13]. Penggunaan aplikasi tidak membutuhkan pekerjaan yang sangat kompleks dari sisi pengguna dalam mengelola websitenya. Kompleks disini dalam artian tidak membutuhkan installasi server kembali dengan kata lain installasi sudah dilakukan di flask.