BAB III

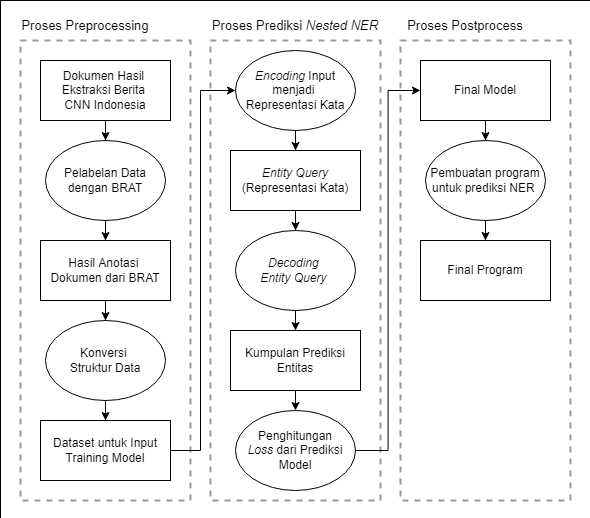
# ARSITEKTUR SISTEM

Bab ini akan membahas tentang arsitektur sistem keseluruhan tugas akhir ini. Arsitektur yang akan dibahas antara lain arsitektur sistem secara utuh, kemudian dibagi menjadi sub-proses tertentu. Pembagian sub-proses ini untuk membantu pengertian dan penjelasan lebih mudah dan detail kepada struktur arsitektur sistem. Pembahasan secara detail untuk masing-masing langkah / proses yang dilakukan dalam arsitektur pada sistem akan dibahas pada bab-bab berikutnya.

## Arsitektur Sistem Umum

Subbab ini akan menjelaskan arsitektur sistem secara keseluruhan selama pengerjaan tugas akhir. Arsitektur sistem ini akan diulas secara detail pada subbab-subbab berikutnya. Subbab arsitektur sistem umum akan membahas aliran data dari sub-proses pertama sampai akhir untuk mengetahui gambaran secara umum sistem ini. Visualisasi arsitektur sistem dapat dilihat dari gambar 3.1 dan dapat dilihat dari gambar tersebut terjadi kurang lebih lima buah proses. Dan untuk setiap beberapa proses telah dikelompokkan menjadi kelompok sub-proses agar dapat dimengerti proses tersebut dilakukan dengan tujuan tertentu. Semisal untuk pra proses, mungkin untuk prediksi NER atau pasca proses untuk programnya.

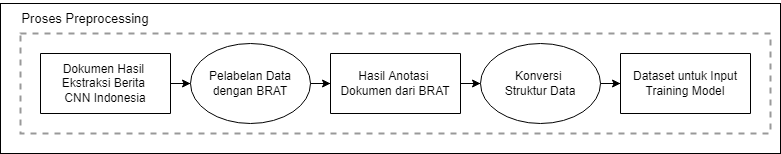
Gambar 3.1 memiliki pembagian tiga sub-proses, pra proses, proses prediksi nested NER, dan pasca proses. Pra proses adalah proses paling pertama, dan memiliki dua proses yang berhubungan dengan dataset. Proses ini fokus pada memroseskan dataset menjadi input yang sesuai untuk model program. Dari dataset mentah dari ekstraksi berita CNN Indonesia, akan diubah menjadi struktur data sesuai dengan struktur untuk model yang digunakan di tugas akhir ini. Kemudian lanjut kepada sub-proses berikutnya adalah proses prediksi nested NER. Proses yang dilakukan pada sub-proses ini adalah bagian dari model. Metode-metode yang ditentukan untuk training model sehingga dapat melakukan prediksi nested NER. Dan metode ini akan dijelaskan lebih detail pada bab-bab berikutnya. Setelah proses paling akhir proses ini, yaitu penghitungan *loss*, proses ini menghasilkan model final yang telah memiliki informasi dan kemampuan untuk menentukan nested NER. Dan model akhir ini akan menjadi input utama dari proses berikutnya yaitu pasca proses. Sub-proses bagian pasca proses ini bertujuan untuk menyiapkan model yang diterima, untuk menerima input dan output yang dapat dilihat oleh user / orang. Untuk detail dari masing-masing arsitektur sistem akan dijabarkan dalam subbab-subbab berikutnya.



Gambar 3.1  
Arsitektur Sistem

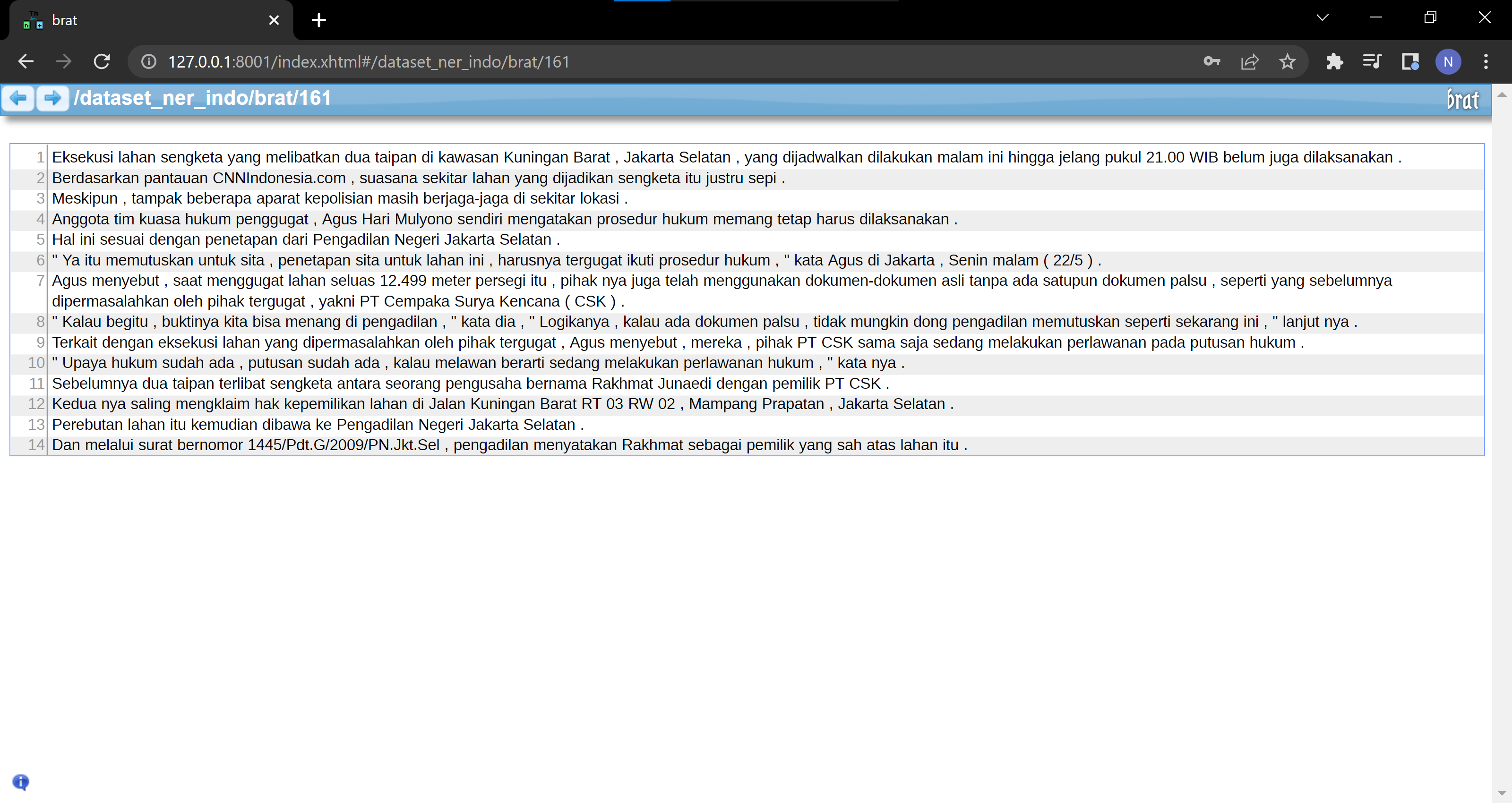
## Arsitektur Sistem dari Pra Proses

Subbab ini akan membahas tahap pra proses yang akan dilakukan dalam arsitektur sistem ini. Pembahasan pada bagian ini akan menjabarkan mengenai proses yang akan dilakukan dalam tahap ini serta input dan output yang akan digunakan dan dihasilkan dalam tahap pra proses ini. Gambaran setiap tahap pra proses dapat dilihat pada gambar 3.2. Dapat dilihat terdapat dua proses (digambarkan dengan lingkaran) dan tiga data (digambarkan dengan persegi panjang) yang akan digunakan dalam proses ini. Setiap data akan diberikan penggambaran dan akan diberikan penjelasan bagaimana data itu akan diterima atau bagaimana akan diolah.



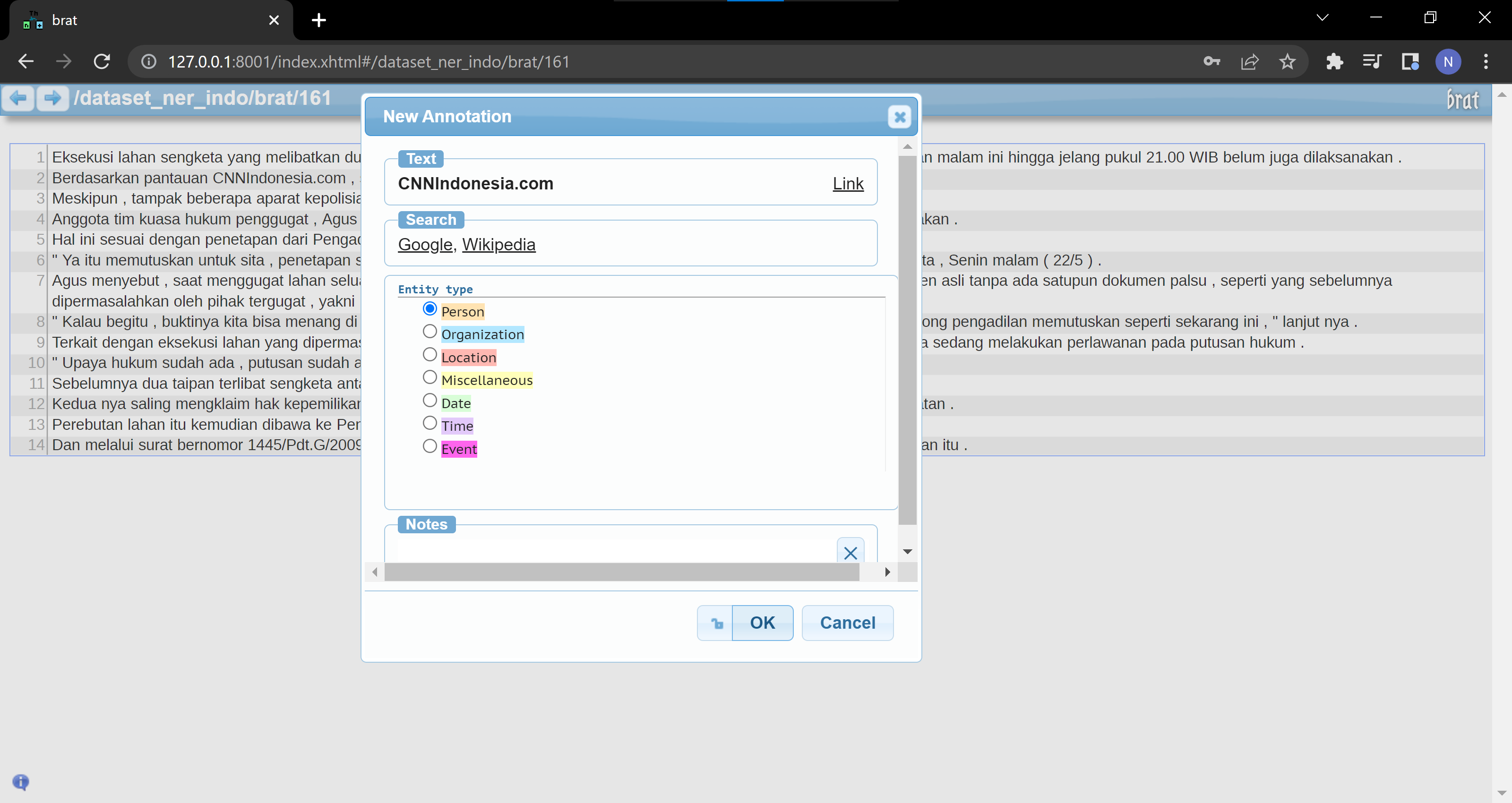
Gambar 3.2  
Arsitektur Sistem Pra Proses

Input pada tahap pertama pra proses adalah hasil ekstraksi berita dari CNN Indonesia. Ekstraksi ini berupa format *text file* (format txt) kemudian data tersebut akan menjadi input untuk proses pelabelan data dengan alat anotasi BRAT. Seperti yang telah dijelaskan, pada bab kedua subbab BRAT, data ini akan diterima dan setiap kalimat akan dipisah dengan fitur *linebreak* dari BRAT sendiri (pemisahan setiap kalimat menjadi 1 baris sendiri dalam sebuah text file). Dan hasil file yang telah diubah ini akan menjadi teks yang muncul pada halaman anotasi BRAT. Isi dari file teks akan dilampirkan dalam halaman BRAT seperti pada gambar 3.3.



Gambar 3.3  
Tampilan File Teks pada BRAT

BRAT dipilih sebagai alat anotasi karena kemudahan kegunaannya dalam pelabelan. Karena penggunaan fitur yang intuitif dan mirip dengan perintah mouse dengan teks pada umumnya, cara pelabelan sebuah / beberapa kata dapat dilakukan dengan menekan dan menggeser *mouse*, memilih kata yang ingin dilabelkan. Kemudian akan muncul *window* jenis-jenis label yang ingin dipilih. Window tersebut juga memiliki fitur lain yang membantu seperti fitur Link untuk menyimpan URL address dari kata yang dianotasikan tersebut untuk menuju ke kata tersebut dengan mudah dan cepat. Fitur search dengan tombol Google atau Wikipedia untuk membantu mencari arti yang relevan dari kata tersebut. Fitur Notes untuk memberi catatan pada anotasi tersebut. Jika selesai melakukan pelabelan atau fitur lain, cukup tekan tombol OK untuk menyimpan hasil perubahan. Tampilan window ini dapat dilihat pada gambar 3.4.

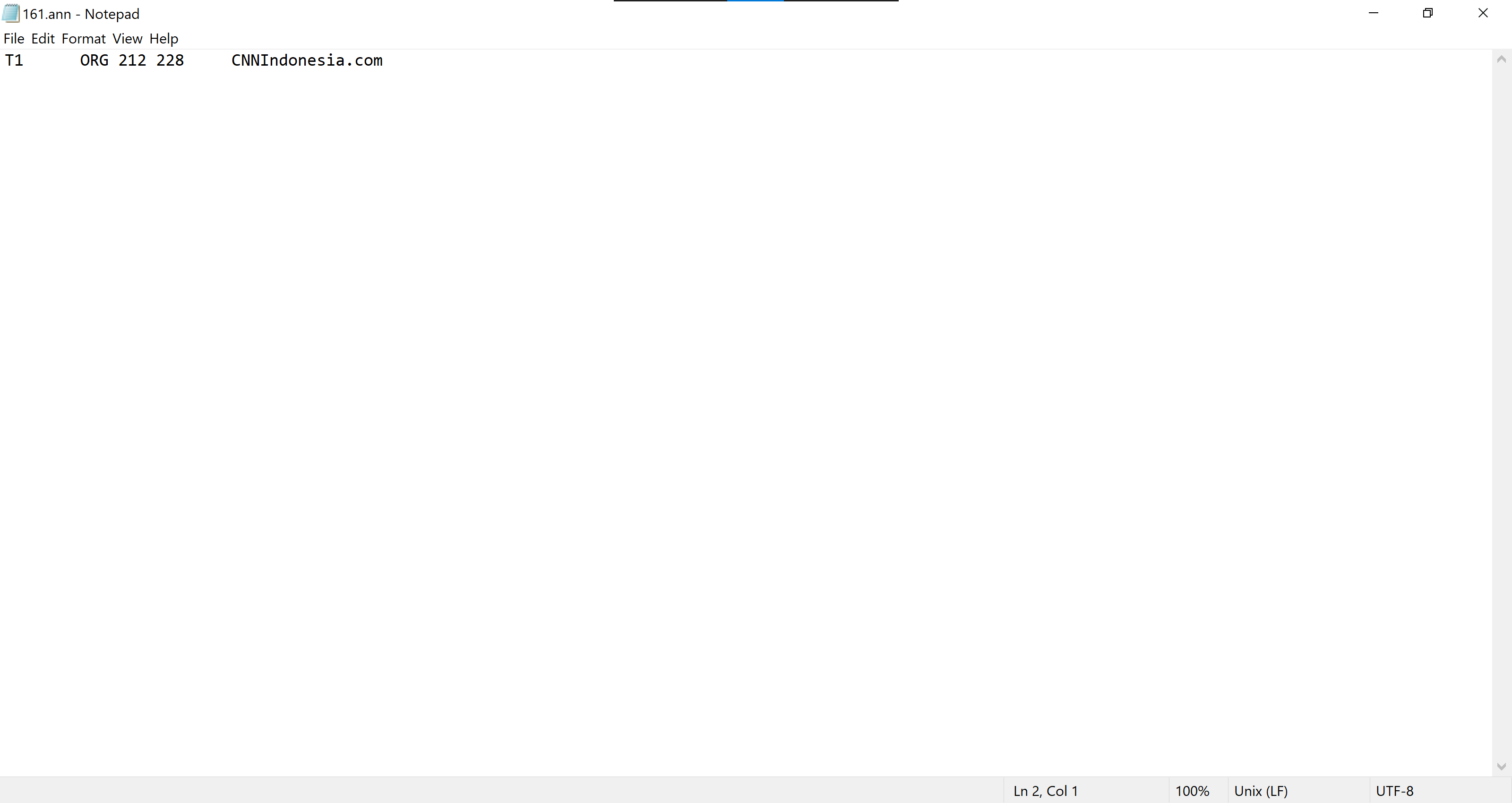


Gambar 3.4  
Tampilan Window Fitur Pelabelan

Contoh pada gambar 3.3 adalah satu file teks/dokumen dari dataset penelitian. Setiap dokumen memerlukan file teks agar dapat dilabelkan, dan untuk pencatatan pelabelan/anotasi yang telah dilakukan akan disimpan dalam jenis file yang dibuatkan sendiri oleh BRAT secara otomatis yaitu jenis file ann (annotation). File teks ini akan menjadi output dari proses pertama yaitu pelabelan data berita CNN Indonesia juga menjadi input untuk proses berikutnya yaitu konversi struktur data. Visualisasi struktur data dari BRAT dan struktur data untuk input training model akan ditampilkan pada gambar 3.5.

Struktur data dari file teks anotasi BRAT memiliki 4 jenis catatan. Gambar dibawah akan diambil sebagai contoh untuk menjelaskan keempat jenis catatan dari BRAT. Catatan terdiri dari T1, ORG, 212, 228 dan CNNIndonesia.com. T1 merupakan kode pelabelan untuk dokumen tersebut, tiap catatan pelabelan akan mendapatkan kode tersebut dan sifatnya unik dan *incremental* (T1, T2, T3, dst). Catatan ORG adalah jenis label yang dipilih, dalam tugas akhir ini, ORG adalah label/jenis entitas organisasi. Angka 212 dan 228 adalah indeks pertama dan terakhir dari dokumen itu untuk mengambil kata/huruf yang dilabel untuk pelabelan tersebut. Sehingga jika dari dokumen diambil kata-kata dari 212 sampai dengan 228 akan mendapatkan kata-kata yang sama dengan catatan berikutnya yaitu CNNIndonesia.com.

Dapat dilihat pada gambar 3.5 (b), jenis filenya struktur data input model bentuk JSON (Java Script Object Notation). Penjelasan atribut dari JSON hanya atribut yang digunakan dalam tugas akhir ini. Tokens adalah *array* kata-kata dalam 1 kalimat yang sedang dianotasikan saat ini. Entities adalah semua label entitas yang ditemukan dalam 1 kalimat itu, dengan bentuk *array of JSONs* yang memiliki 3 atribut. Atribut start dan end adalah index awal dan akhir dari kalimat saat ini, dan type sebagai catatan label/jenis entitas apa untuk kata tersebut. Meskipun tugas akhir ini memiliki batasan tidak menggunakan POS Tag, namun model tetapi meminta POS Tag untuk mengolah data. Karena itu atribut pos itu akan berisi token *unknown* hanya sebagai data pengisi agar tidak memunculkan *error* dalam training program. Ltokens dan rtokens memiliki tujuan yang sama dengan tokens perbedaannya adalah ltoken merupakan kalimat sebelum kalimat saat ini. Dan rtokens adalah kalimat setelah kalimat saat ini.



**(a)**



(b)

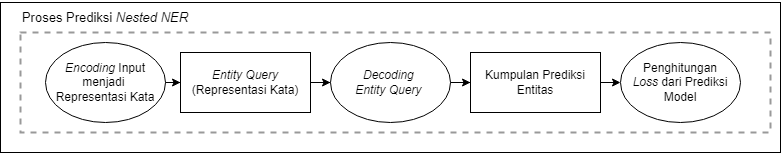
Gambar 3.5  
Struktur Data (a) Data Anotasi BRAT (b) Data Input Model

Dalam tugas akhir ini telah dibuatkan program untuk merubah struktur data BRAT menjadi struktur data Sequence-to-Set, karena tidak adanya program konversi tersebut. Setelah struktur data dari file teks BRAT diubah menjadi struktur data yang dibutuhkan, menghasilkan dataset dengan struktur sesuai untuk menjadi input training model. Dengan ini, penjelasan persiapan dataset dalam bagian pra proses selesai dan dataset tersebut akan dilanjutkan kepada proses berikutnya yaitu proses prediksi nested NER.

## Arsitektur Sistem dari Proses Prediksi Nested NER

Subbab ini menjelaskan proses prediksi nested NER dengan melakukan training dengan metode Sequence-to-Set Network. Gambar 3.6 menggambarkan visualisasi aliran system untuk prediksi nested NER ini. Terdiri dari tiga proses dan dua data yang digunakan. Untuk detail seperti metode apa yang digunakan dalam tiap proses, modifikasi atau parameter yang digunakan dalam tiap proses tidak akan dijelaskan dalam bab ini namun pada bab-bab kedepannya.

Aliran sistem ini akan dimulai dengan sebuah proses yang menerima input dari proses sebelumnya yaitu tahap pra proses. Input merupakan dataset yang sudah diubah sesuai struktur data yang ditentukan oleh model Sequence-to-Set Network. Proses ini mengambil inputnya dan melakukan encoding, hal ini bertujuan untuk membuat representasi kata yang baru dengan informasi yang terdapat dari input dataset saat itu. Hal ini dicapai dengan bantuan berbagian jenis embedding yang saling digabungkan. Representasi kata yang baru ini dibutuhkan agar membantu computer untuk training model dengan representasi dalam bentuk yang dapat mudah diolah yaitu angka bukan kata-kata.



Gambar 3.6  
Arsitektur Sistem Proses Prediksi Nested NER

Representasi kata tersebut merupakan output dan juga input pertama dari sub-proses ini, dimana representasi kata itu memiliki nama atau sebutan *Entity Query/Queries*. Entity Queries ini, seperti yang dijelaskan sebelumnya, merupakan representasi kata-kata dari input dataset dalam bentuk angka agar lebih mudah diproses oleh model. Secara teori, isi dari entity queries ini adalah gabungan berbagai jenis embedding yang ada. Jenis embedding ini akan dijelaskan pada sub-bab berikutnya. Dan data ini, dari urutan input, memiliki panjang *l x d*, dimana *l* adalah panjang urutan input saat ini, dan *d* sebagai dua kali panjang *hidden size* dari LSTM yang digunakan pada proses encoding (proses sebelumnya).

Lanjut pada proses berikut yang menerima entity queries adalah bagian kedua dari arsitektur sistem dari model ini yaitu proses *decoding* entity queries. Target dari proses ini adalah untuk melakukan membaca informasi yang telah “diringkas” dalam entity queries. Sebagian besar dari pekerjaan *decoder* dilakukan dengan mekanisme Attention yang diambil dari metode Transformer (metode ini telah dijelaskan pada bab sebelumnya mengenai bagian decoder). Decoder akan mempelajari kata-kata yang perlu diperhatikan dan ketergantungan antar entitas untuk mengetahui pola prediksi nested NER. Setelah entity queries dilewatkan self-attention dan cross-attention, hasil dari cross-attention akan melewati bagian Feed Forward Network (FFN), bagian ini bertugas untuk lebih mengetahui hubungan antar nilai dalam embedding. Dan output dari FFN akan menjadi input pada layer terakhir bagian decoder yaitu Multilayer Perceptron (MLP) untuk mengklasifikasikan embedding FFN menjadi hasil akhir batasan kiri dan kanan prediksi jenis entitas dan juga jenis entitas yang diprediksikan.

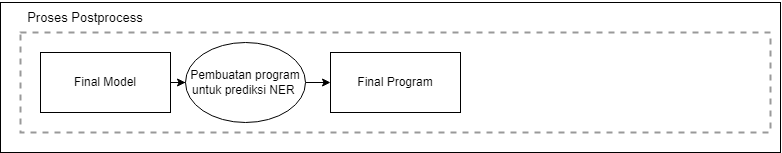
Output dari MLP ini adalah data “kumpulan prediksi entitas” dalam gambar 3.6. Dan dengan output ini akan memasuki proses paling akhir yaitu penghitungan *loss* untuk menentukan performa dari model yang sedang training. Hal ini dilakukan dengan metode *bipartite matching*. Suatu metode yang sering digunakan untuk membantu penghitungan *assignment matching*. Dalam bagian ini, akan diusahakan untuk mencari nilai loss, dibutuhkan nilai optimal pasangan prediksi entitas dengan target entitas sebenarnya. Nilai optimal tersebut diambil dari algoritma Hungarian. Nilai loss ini akan membantu model untuk mempelajari cara prediksi nested NER.

Hasil dari proses prediksi ini adalah model yang sudah melewati proses training dan memiliki ilmu untuk prediksi nested NER dengan optimal. Setelah mendapatkan model yang optimal, model ini diperlukan untuk memprediksikan input kalimat yang ingin dicarikan entitas nya. Karena itu perlu dilewatkan tahap pasca proses yang menyiapkan model untuk menjadi program yang dapat digunakan user dengan mudah.

## Arsitektur Sistem dari Proses Postprocess

Proses pasca dari bagian prediksi akan dibahas dalam bagian subbab ini. Alur dari proses tersebut dapat dilihat pada gambar 3.7 bahwa hanya terdapat satu proses yang dijalankan dengan dua hasil proses yang disediakan. Final model merupakan hasil akhir model dengan metode sequence-to-set network yang telah ditraining sedemikian rupa dengan akurasi terbaik. Model ini akan berbentuk file ekstensi .bin dan beberapa file lainnya yang akan menjadi file pendukung untuk menjalankan model tersebut. File tersebut adalah config.json, extra.state, pytorch\_model.bin, special\_tokens\_map.json, tokenizer\_config.json, vocab.txt.

Config.json adalah konfigurasi untuk model tersebut saat akan dijalankan untuk prediksi dari input program. Extra.state adalah metadata untuk model mengenai versi model dan data lainnya. Pytorch\_model.bin adalah model tersebut yang akan dijalankan dengan bantuan informasi lain seperti konfigurasinya (config.json), token spesial yang akan digunakan dalam proses analisa dataset (special\_tokens\_map.json), juga kosakata dari kata-kata yang pernah muncul didataset (vocab.txt). File tokenizer\_config.json adalah file berisi informasi mengenai tokenizer model yang digunakan untuk perubahan kata-kata dari dataset menjadi token sesuai dengan konfigurasi yang ditentukan. Contoh konfigurasi adalah apakah semua kata-kata dikonversikan menjadi *lower case*, nama dari model tokenizer yang digunakan, token-token spesial yang dapat digunakan tokenizer untuk kata/karakter spesial (kata-kata yang tidak diketahui diberikan *unknown token* biasanya dalam bentuk [UNK]/<UNK>. Seluruh file ini akan digunakan dalam proses berikutnya yaitu prembuatan program prediksi NER dengan final model.



Gambar 3.7  
Arsitektur Sistem Pasca Proses

Proses pembuatan program untuk prediksi NER adalah proses dimana file-file dari model yang telah melewati proses training, akan ikut serta dalam pembuatan program baru. Program ini akan menerima input user, kemudian program akan memberikan input ini kepada final model untuk memprediksi tag named entity dari kalimat input user tersebut. Program ini jika selesai dapat digunakan untuk menerima input beberapa kalimat user (3-4 kalimat), kemudian akan dioutput kan dalam file bentuk .ann, seperti yang pernah disebut, file pelabelan untuk BRAT untuk mempermudah visualisasi.

Program ini akan terdiri dari beberapa file .py (file bahasa pemrogramman Python), dimana file-file tersebut juga memiliki alur sendiri. Pertama program menerima input kalimat dalam bentuk txt, kemudian kalimat tersebut akan dikonversikan menjadi struktur data yang sesuai untuk diterima model agar dapat diprediksi. Setelah dikonversikan, program akan melempar kalimat yag sudah dikonversi kedalam model yang otomatis akan memberikan sejumlah entitas prediksi yang diprediksikan. Program akan mengubah hasil prediksi ini menjadi file .ann agar dapat melihat visualisasi hasil prediksi tersebut dalam program BRAT.