# Kelompok:

Febrian Nugroho / 2301930551

Felix Filipi / 2301877590

# Pengembangan Chatbot Wikipedia Berbasis Encode-Decode Menggunakan Algoritma LSTM

## Latar Belakang

Perkembangan teknologi yang sangat cepat telah membawa banyak perubahan dalam kehidupan manusia. Perkembangan yang sangat cepat ini muncul akibat teknologi-teknologi baru yang telah merevolusi berbagai sektor, baik di bidang pendidikan maupun di bidang industri. Salah satu teknologi yang telah merevolusi industri tersebut adalah *Artificial Intelligence (AI)* [1]. *Artificial Intelligence (AI)* adalah salah satu bidang ilmu komputer yang berkembang sangat cepat, yang mana pengimplementasiannya dapat diterapkan di berbagai bidang. Hasil pengembangan dari teknologi *Artificial Intelligence* sendiri ada banyak, salah satunya adalah *chatbot* yang merupakan tema pada penelitian kami pada kesempatan kali ini.

Pengembangan *chatbot* dalam industri 4.0 ini sedang mengalami peningkatan yang sangat signifikan [2]. Perkembangan ini telah menjadi salah satu fokus utama dalam automasi industri abad ini, hal ini dapat dilihat pada berubahnya sistem *Question and Answer (QnA)* dari yang semula manual dengan menggunakan manusia sebagai aktor, menjadi *chatbot* sebagai aktor penggantinya. Dan tidak hanya itu saja, kedepannya perkembangan ini akan menjadi semakin pesat mengingat prospek dari *chatbot* ini sangat tinggi untuk diimplementasikan pada perusahaan dibandingkan dengan menggunakan sistem *QnA* tradisional yang membutuhkan biaya serta sumber daya yang cukup mahal dalam penggunaanya [3].

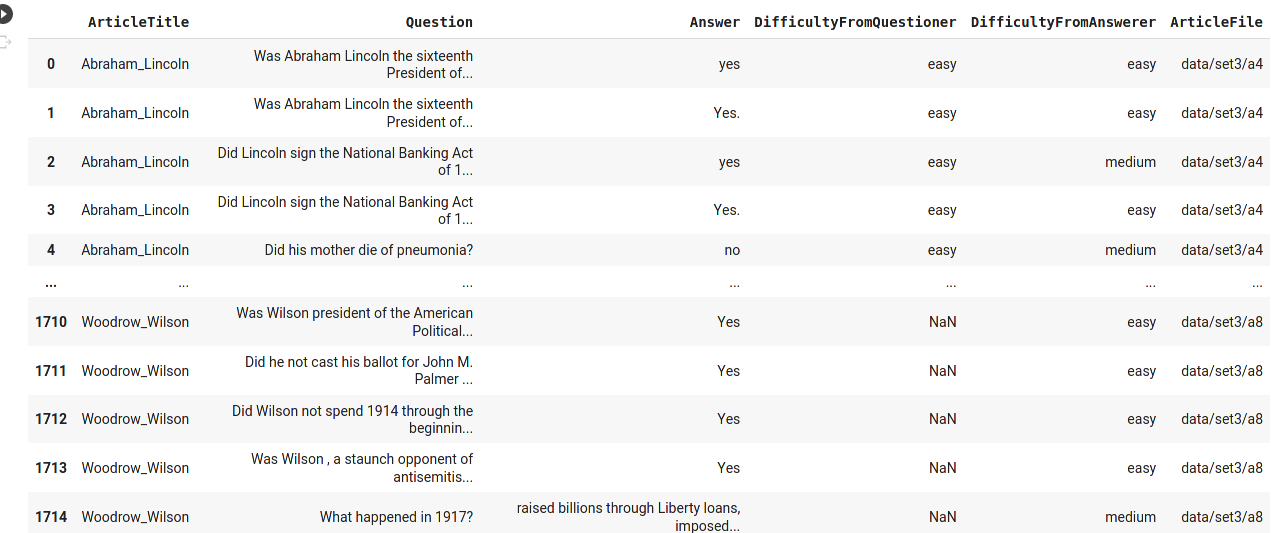
Melihat peluang tersebut, kami melakukan penelitian dalam mengembangkan sebuah *chatbot* yang dapat menerapkan sistem *QnA* berbasis pertanyaan-pertanyaan yang diambil dari *corpus wikipedia*. Penelitian ini menggunakan konsep *encoder* dan *decoder* dengan menggunakan algoritma *Long-Short Term Memory (LSTM)* pada pengembangannya. Dengan adanya penelitian ini diharapkan, kami dapat menguji efektifitas dari algoritma yang digunakan dan dapat mengimplementasikan aplikasi yang telah dibuat pada sektor yang membutuhkan.

## Tujuan dan Manfaat

Tujuan pengembangan aplikasi ini adalah untuk menguji tingkat efektivitas algoritma yang digunakan dalam mengembangkan sistem *chatbot* yang digagas, sehingga dapat digunakan untuk mempermudah pengguna dalam menjawab berbagai pertanyaan yang mereka ajukan.

## Data

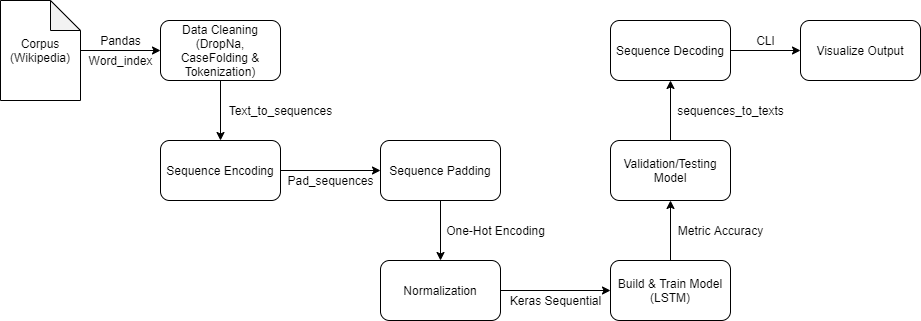
*Corpus* yang digunakan didapatkan dari sebuah situs penyedia *corpus wikipedia* pada link <https://www.cs.cmu.edu/~ark/QA-data/>. Pada *corpus* tersebut, dari 3 buah *dataset* yang tersedia, kami hanya menggunakan satu buah *dataset*, yaitu *dataset* pada *folder* ‘S08’, dikarenakan data yang ada di *corpus* itu sangatlah besar, dan kemampuan komputasi dari perangkat yang kami gunakan cukup terbatas. Berikut merupakan gambaran *raw data* dari *dataset* yang kami gunakan.

**Gambar 1**. Raw Data

Data tersebut mempunyai struktur berupa 1714 baris data dan 6 kolom yang terdiri dari ArticleTitle, Question Answer, DifficultyFromQuestioner, DifficultyFromAnswerer, ArticleFile.

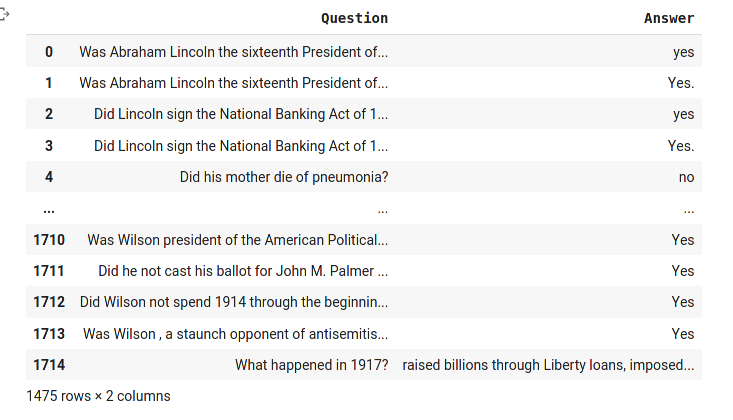
## Metodologi Penelitian

Sebelum masuk lebih jauh pada penjelasan mengenai penelitian kami, kami telah membuat diagram alur untuk memudahkan pemahaman tentang proses penelitian kami:



**Gambar 2**. Tahapan penelitian

1. *Corpus.*

Corpus yang kami gunakan berasal dari *dataset wikipedia*, yang dapat diambil dari [https://www.cs.cmu.edu/~ark/QA-data](https://www.cs.cmu.edu/~ark/QA-data/)/. Kami akan melakukan *feature selection* pada *corpus* ini sehingga *corpus* tersebut diubah menjadi 2 kolom, yaitu kolom ‘*Question’* yang berperan sebagai *feature* dan kolom ‘*Answer’* yang berperan sebagai *target*. Berikut gambaran dari *dataset* yang digunakan.

**Gambar 3**. Corpus penelitian

1. *Data Cleaning.*

Data dari *corpus* yang digunakan masih belum dapat diproses ke dalam model, sehingga kita perlu untuk melakukan *data cleaning* terlebih dahulu. Dengan melakukan *data cleaning,* *corpus* yang sebelumnya masih memiliki banyak *bias* dapat dibuat menjadi efektif sehingga dapat meningkatkan tingkat akurasi dari model yang dibuat.

Pada penelitian kali ini, *data cleaning* akan dilakukan dengan 3 cara, yaitu.

* Melakukan *drop* pada *null* dan *duplicate data*, dengan menggunakan fungsi *dropna*, dan *drop\_duplicates* dari *library pandas*.
* *Case Folding*, yaitu dengan menghilangkan tanda baca yang seringkali tidak diperlukan karena tidak akan menambah nilai pada suatu model NLP (*Punctuation*), Mengubah text menjadi *lowercase*, dan juga menghapus *whitespace*. *Case folding* ini dilakukan dengan membuat fungsi secara manual dengan menggunakan *regular expression*.
* *Tokenizing*, yaitu dengan memisahkan teks menjadi potongan-potongan data yang dikenal sebagai token menggunakan fungsi *word\_index* dari *library keras tokenizer.*

1. *Sequence Encoding.*

Pada tahapan ini, data yang sudah di *cleaning* akan di *encode* / disandikan menjadi sebuah urutan / *sequences*. *Sequences Encoding* ini dapat dilakukan dengan pembuatan *word\_index*, yang bertujuan untuk memetakan/mengindexkan setiap kata pada kalimat ke representasi numeriknya. Proses ini juga dikenal sebagai proses *word to index (word2idx)* yang sangat penting untuk pemetaan tiap kalimat.

Setelah melakukan pemetaan dan menandai data (*word2idx*), kita akan melakukan *encode*/penyandian dari setiap kata pada urutan data tersebut. Sebagai contoh, pada kalimat “Pengembangan chatbot wikipedia berbasis LSTM”, kita akan mengubahnya menjadi seperti “6 9 4 21 5 0 0 dst”, yang mana masing-masing angka akan dicocokan dengan indeks pada kata-kata yang sesuai.

*Sequence encoding* ini diperlukan karena *neural network* pada model kita hanya bekerja pada perhitungan angka, sehingga apabila kita meneruskan nilai tersebut dalam bentuk kata tanpa diubah ke bentuk angka seperti diatas, maka hal tersebut tidak akan bekerja. Untuk melakukan proses *Sequence Encoding*, kita dapat menggunakan fungsi dari *library keras* yang bernama *texts\_to\_sequences* yang dapat membantu mengubah sebuah text menjadi *sequences* yang diinginkan dengan mudah.

1. *Sequence Padding.*

*Sequence padding* disini dibutuhkan untuk mencapai panjang maksimal dari *sequences* untuk mengisi kalimat yang di *encode*. Hal ini dapat dicapai dengan menambahkan *padding* berupa tambahan angka 0 di akhir *sequences*. Penambahan angka ini digunakan untuk mengisi nilai kosong pada urutan yang ada, sehingga semua nilai memiliki panjang vektor yang sama. Selain itu, jika panjang dari *sequences* melebihi panjang maksimal yang telah ditentukan, maka kita akan memotong *sequences* tersebut sesuai dengan panjang maksimal yang telah ditentukan sebelumnya.

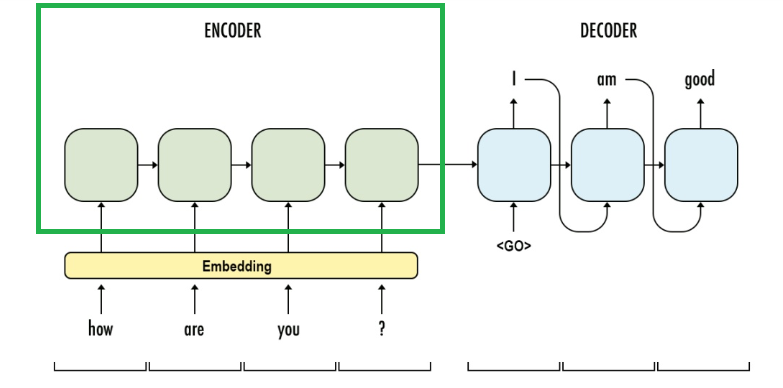
1. *Normalization.*

Data dari *Sequences* yang sudah di *encode* menjadi nilai numerik perlu untuk dinormalisasi menggunakan *One-Hot Encoder*, untuk mengubah masing-masing dari angka yang di *encode* menjadi nilai berupa vektor 0 dan 1. Penggunaan *One-Hot Encoder* disini bertujuan untuk memudahkan dalam pemrosesan *data training* sehingga diharapkan dapat meningkatkan tingkat akurasi dari model yang telah dibuat.

1. *Build & Train Model*

Setelah data telah di *preprocessing* dengan berbagai langkah di atas, kita akan memasukkan data tersebut ke dalam sebuah model *neural network* yang bertujuan untuk memprediksi jawaban dari pertanyaan yang telah dibuat. Untuk melaksanakan hal ini, kita perlu untuk mendefinisikan terlebih dahulu algoritma yang akan digunakan. Pada kasus kami, kami menggunakan algoritma LSTM sebagai arsitektur dari model yang digunakan.

Pada penelitian kali ini, kami membuat model yang dimaksud dengan bantuan *library Keras* dari *Tensorflow* dengan menggunakan 128 layer *encoder-LSTM*, yang mana output dari proses ini digunakan untuk menyimpan nilai *state* dari LSTM, untuk diproses pada layer selanjutnya yaitu pada 128 layer *decoder-LSTM*, yang akan dikoneksikan menjadi *fully-connected layer* menggunakan *dense layer* sebanyak output dari data yang di *input*. Untuk fungsi aktivasi yang digunakan pada penelitian ini adalah fungsi *softmax* dan *loss function* berupa *categorical cross-entropy* dikarenakan input yang digunakan telah diproses menggunakan *one-hot encoder.* Dan berikut merupakan gambaran proses dari model yang telah kami buat.

**Gambar 4**. Gambaran Model

Setiap data training yang telah di *preprocess* akan dimasukkan ke dalam 128 layer *encoder-LSTM* dan hasil dari proses ini akan digunakan sebagai *input* pada *decoder-LSTM* sehingga menghasilkan *output* sesuai dengan jumlah target yang diminta.

1. *Validation & Testing Model*

Pada tahapan ini, model yang sudah dibuat akan divalidasi menggunakan dua metode *validation*, yaitu *validation accuracy* dan *validation loss*. *Validation accuracy* bertujuan untuk menampilkan akurasi yang dicapai saat proses *training* berlangsung. Sementara itu, *validation* *loss* bertujuan untuk menampilkan kesalahan prediksi / *loss* pada saat *training* berlangsung. *Validation* ini digunakan untuk memastikan tingkat akurasiprediksi serta kesalahan/*loss* yang dikeluarkan. Selain itu, kita juga melakukan *testing* dengan menggunakan set uji coba *non-training* yang telah kami tetapkan, untuk memastikan keakuratan dari model yang telah dibuat.

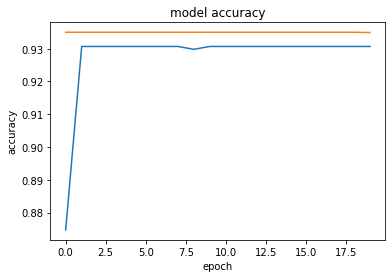
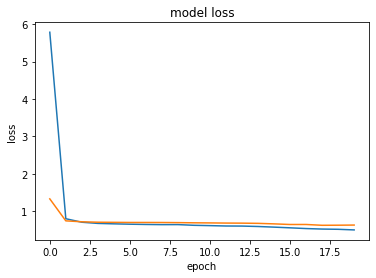
1. *Sequences Decoding*

Setelah proses prediksi dari model *deep learning* berhasil, kita perlu memetakan kembali nilai dari bentuk angka menjadi bentuk kata/*string* dengan melakukan *sequences decoding*. Proses ini diperlukan untuk membaca *output* dari model yang telah dibuat, sehingga memudahkan pembaca dalam menggunakan aplikasi yang dibuat. Proses ini dapat dilakukan dengan bantuan fungsi dari *library keras* yaitu fungsi *sequences\_to\_texts* yang juga dikenal sebagai proses *index to word (idx2word)*.

1. *Visualize Output.*

Pada tahapan ini, hasil output yang telah dihasilkan akan divisualisasikan menggunakan bentuk *Command Line Interface (CLI)* yang bertujuan untuk menjaga stabilitas serta efisiensi dari produk ini.

## Hasil Eksperimen dan Analisis

**Gambar 5.** *Validation Accuracy* **Gambar 6.** *Validation Loss* 

Hasil eksperimen yang diberikan adalah hasil dari uji *validation loss* dan juga *validation accuracy* dari *model training* yang sudah dibuat. Dapat dilihat dari grafik diatas, hasil dari kedua validasi menunjukkan tingkat akurasi yang cukup tinggi dan juga *loss* yang cukup rendah, dimana hasil akhir dari *accuracy* adalah 93.19% dan *validation accuracy* adalah 93.49%, sedangkan untuk *loss* adalah 0.4836 dan *validation loss* adalah 0.6246. Sehingga dapat dikatakan bahwa *model training* yang dibuat cukup baik. Hasil ini juga menunjukkan bahwa langkah *preprocessing* yang sudah dilakukan cukup efektif dalam meningkatkan tingkat keakuratan model yang dibuat.

Setelah kita melakukan uji validasi pada *data training*, langkah selanjutnya adalah melakukan *testing* pada model yang telah dibuat untuk memastikan keakuratan sistem pada *dataset* yang belum dikenali oleh model sebelumnya. Untuk melakukan *testing* ini, kami membuat *test set* berupa 5 buah pertanyaan untuk diuji pada model yang sudah dibuat, yang mana ke 5 buah pertanyaan ini dapat dilihat pada tabel dibawah.

|  |  |
| --- | --- |
| Question | Answer |
| Are beetles insect | Yes |
| who montevideo | The Spanish |
| what turtle is extinct | Paracryptodira |
| what ghana mean | warrior king |
| where finland located | northern europe |

**Tabel 1.** *Test Set*

Kami melakukan *testing* pada *test set* diatas, namun sayangnya kami menemukan terjadinya *error* dimana model mengalami *overfit* sehingga memberikan hasil yang kurang memuaskan yakni sebesar 40%, yang mana berarti model hanya mampu menjawab 2 pertanyaan secara tepat dari 5 pertanyaan yang diberikan. Setelah melakukan berbagai upaya kalibrasi *hypertuning*, sayangnya dari data tersebut masih belum ditemukan langkah yang tepat untuk memperbaiki *overfit* ini. Hal ini mungkin terjadi akibat dataset yang digunakan untuk training terlalu banyak, atau mungkin juga akibat *human-error* pada saat melakukan pemrograman. Kedepannya kami akan melakukan evaluasi lebih lanjut untuk memperbaiki serta merevisi laporan penelitian yang telah dibuat kali ini, dan memberikan klarifikasi atas hasil yang telah dibuat.

## Simpulan

Setelah melihat hasil dari proses *training* dan *testing*, maka kami telah membuat kesimpulan sementara, dimana model *encoder-decoder LSTM* kurang memadai untuk dijadikan model dalam pengembangan aplikasi *chatbot*. Hal ini kemungkinan terjadi akibat intensitas jumlah data yang digunakan selama training yang cukup besar, atau mungkin terjadi akibat adanya *human-error* selama peneliti melakukan pemrograman. Maka dari itu, kedepannya kami akan memberikan laporan lebih lanjut untuk mengklarifikasi hasil dari laporan yang telah dibuat, sehingga dapat memastikan kebenaran serta fakta dari laporan kami kali ini.

## Referensi

[1] B. Setiaji and F. W. Wibowo, “Chatbot Using a Knowledge in Database: Human-to-Machine Conversation Modeling,” Proc. - Int. Conf. Intell. Syst. Model. Simulation, ISMS, vol. 0, no. January 2016, pp. 72–77, 2016, doi: 10.1109/ISMS.2016.53.

[2] R. N. Astuti and M. Fatchan, “PERANCANGAN APLIKASI TEKNOLOGI CHATBOT UNTUK INDUSTRI KOMERSIAL 4.0,” vol. 0, no. September, pp. 339–348, 2019.

[3] E. L. Amalia and D. W. Wibowo, “Rancang Bangun Chatbot Untuk Meningkatkan Performa Bisnis,” J. Ilm. Teknol. Inf. Asia, vol. 13, no. 2, p. 137, 2019, doi: 10.32815/jitika.v13i2.410.

[4] [1] N. A. Smith, M. Heilman, and R. Hwa, “Question generation as a competitive undergraduate course project,” Proc. NSF Work. Quest. Gener. Shar. Task Eval. Chall., pp. 4–6, 2008, [Online]. Available: <http://www.cs.cmu.edu/~nasmith/papers/smith+heilman+hwa.nsf08.pdf%0Ahttp://www.cs.cmu.edu/~nasmith/papers/smith+heilman+hwa.nsf08.pdf?t=1&cn=ZmxleGlibGVfcmVjc18y&refsrc=email&iid=2bf1bbb68ba140108d8b5fbe61f78637&uid=58984577&nid=244+272699400%0Ahttps://ww>.

[5] M. Mayo, “Tokenization and Text Data Preparation with TensorFlow & Keras,” KDnuggets, 2020. <https://www.kdnuggets.com/2020/03/tensorflow-keras-tokenization-text-data-prep.html>.

[6] A. Sojasingarayar, “Seq2Seq AI Chatbot with Attention Mechanism,” no. May, pp. 1–18, 2020, [Online]. Available: <http://arxiv.org/abs/2006.02767>.