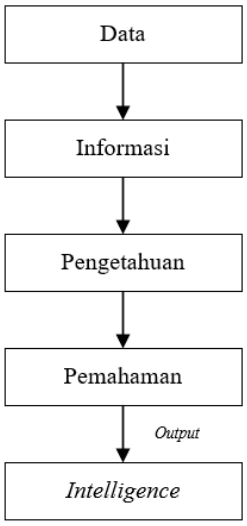
BAB II
TINJAUAN PUSTAKA

2.1 *Artificial Intelligence*

Artificial Intelligence atau AI dapat didefinisikan sebagai suatu mesin atau alat pintar yang dapat berpikir lebih cerdas dan membuat mesin lebih berguna [17]. AI bertujuan untuk mengotomatisasi aktivitas yang saat ini membutuhkan kecerdasaan manusia. AI sering digunakan dalam bidang pendidikan, kesehatan, ekonomi, dan pertanian [18]. *Artificial Intelligence* atau kecerdasan buatan muncul sejak pembuatan komputer modern pada tahun 1940 dan tahun 1950. Perkembangan AI dalam teknologi dan informasi sangat cepat dan pesat. AI kini sudah mempengaruhi kehidupan manusia, AI sudah menjadi sebuah kebutuhan dalam menjalankan kehidupan sehari-hari. Proses bekerjanya sebuah AI ialah menggabungkan data secara cepat, dilanjutkan dengan pengolahan data yang dilakukan secara berulang, dan perancangan algoritma cerdas [19]. Pola kecerdasan buatan dapat diperlihatkan pada Gambar 2.1 di bawah ini.



Gambar 2.1 Pola Kecerdasaan Buatan

Berdasarkan Gambar 2.1 di atas kercerdasaan manusia berasal dari perilaku manusia yang dapat ditelusuri mulai dari kombinasi unik genetika, pola pengasuhan, dan paparan individu terhadap berbagai situasi dan lingkungan sekitar. Kecerdasaan manusia dapat beradaptasi dengan lingkungan baru. Manusia mengandalkan daya komputasi, memori, dan kemampuan berpikir otak, sebaliknya AI mengandalkan data dan instruksi yang dimasukkan ke dalam sistem. Secara fungsional, manusia membutuhkan waktu yang lama dalam memproses dan memahami masalah. Namun, AI dapat membantu manusia untuk memberikan hasil yang cepat dan akurat.

2.2 *Machine Learning*

Machine Learning merupakan cabang dari kecerdasan buatan yang digunakan untuk belajar dan meningkatkan kinerja dengan menggunakan pengalaman (*history*) atau data yang diberikan. *Machine learning* memberikan kemampuan pada komputer untuk belajar dari data yang diberikan dan mempelajari pola serta informasi yang tersembunyi dalam data yang diberikan. hal tersebut dilakukan untuk membuat prediksi atau mengambil keputusan tanpa harus secara eksplisit [20].

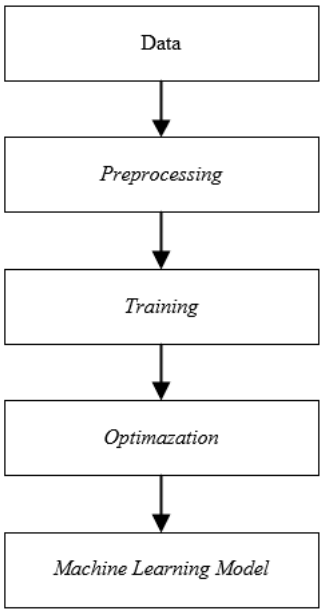
Machine learning dapat dikelompokkan berdasarkan proses pembelajarannya menjadi tiga jenis yaitu.

1. *Supervised learning* atau pembelajaran terarah yang melibatkan penggunaan data yang telah diberikan sebuah label. Label merupakan jawaban atau klasifikasi yang benar dari setiap sampel data pada dataset. Model dilatih untuk menemukan pola dalam data yang dapat ditarik untuk menghasilkan sebuah prediksi pada data yang baru. Contoh dari penggunaan algoritma *supervised learning* ialah regresi *linear*, *naïve bayes, k-nearest neighbors* atau KNN, *decision trees*, dan lain-lain.
2. *Unsupervised learning* atau pembelajaran tanpa pengawasan merupakan algoritma yang tidak menggunakan label disetiap data. Penggunaan *unsupervised learning* bertujuan untuk mencari pola, kelompok, struktur pada data tanpa melabeli data tersebut. Contoh penggunaan algoritma

unsupervised learning ialah *k-means clustering, hierarchical clustering*, dan lain-lain.

3. *Reinforcement learning* atau pembelajaran dengan penguatan merupakan pembelajaran melalui interkasi dengan lingkungan. Model yang dihasilkan diuji coba melalui berbagai tindakan dan menerima umpan balik atau *feedback* berupa penghargaan atau hukuman dari lingkungan. Tujuan utama algoritma ini digunakan ialah memaksimalkan keuntungan dalam waktu jangka panjang.

Selain jenis-jenis di atas terdapat metode yang dapat digunakan seperti *transfer learning*. *Transfer learning* menggunakan pengetahuan dari sebuah tugas untuk membantu tugas lainnya. Selanjutnya terdapat *ensemble learning* yaitu sebuah teknik menggabungkan prediksi dari beberapa model, *active learning* merupakan sebuah interaksi antara manusia dan model untuk mengoptimalkan kinerja model [21]. Diagram *machine learning* dapat dilihat pada Gambar 2.2 sebagai berikut.



Gambar 2.2 *Machine Learning* Diagram

Berdasarkan Gambar 2.2 di atas *machine learning* membutuhkan data yang dijadikan sebagai *input* untuk melatih model. Proses selanjutnya terjadi pembersihan data, transformasi dan penyusunan data sesuai dengan kebutuhan model, kegiatan ini dinamakan *preprocessing*. Model yang sudah ada melakukan pelatihan yang sudah di-*setting* sebelumnya. Model diberikan data pelatihan untuk menyesuaikan dengan parameter atau bobot sehingga hasil proses *training* ini dapat lebih maksimal. *Optimazation* ialah proses *training* yang dilakukan berulang untuk menghasilkan evaluasi dalam bentuk parameter tertentu [22]. Model *machine learning* ialah sebuah algoritma yang dihasilkan dalam proses pelatihan sehingga model tersebut dapat dilakukan *test* data untuk mengetahui tingkat akurasi serta optimalisasi dari algoritma yang sudah di buat.

Pada penelitian ini, jenis pembelajaran yang digunakan ialah *supervised learning* atau pembelajaran terarah. Metode USE dimana model dilatih menggunakan data yang sudah dilabeli dan dilatih untuk mempelajari pola dalam data sehingga hasil akhir yang dapat ditarik menggambarkan hubungan antar data yang diberikan. Pada metode TFRS juga termasuk menggunakan *supervised learning* dimana model dilatih menggunakan data yang sudah dilabeli dan melakukan proses pembelajaran untuk mengetahui pola yang ada pada data. Sehingga, menghasilkan sebuah sistem rekomendasi yang sesuai dengan preferensi pengguna.

2.3 Pengolahan data

Pengolahan data merupakan proses memanipulasi, mentransformasi, dan menganalisis data guna mendapatkan informasi yang diperlukan. Proses pengolahan data memalui beberapa rangkaian yaitu untuk membersihkan, mengorganisir, mengubah, dan menganalisis data untuk mendapatkan informasi dengan tujuan tertentu [23]. Berikut ini penjelasan lebih rinci mengenai serangkaian proses pengolahan data.

1. Pengumpulan data menjadi langkah pertama yang dilakukan untuk mengelola sebuah data. Data dikumpulkan dari berbagai sumber. Data bisa

berbentuk karakter seperti alfabet dan angka. Susunan dari data dimulai dari *bits*, *bytes*, *fields*, *records*, *file*, dan *database* [24].

2. Pembersihan data dilakukan setelah semua data yang diperlukan sudah terkumpul. Pada proses identifikasi dan penanganan *missing value* (nilai yang hilang atau kosong), *outlier* (nilai yang ekstrem), dan kesalahan pada data, serta ketidak konsistenan data. Pembersihan data menghapus data-data yang tidak relevan [25].
3. Transformasi data dilakukan untuk mengubah format data sesuai dengan proses analisis yang dilakukan. Pada proses ini mengimplikasikan normalisasi data, pengkodean kategori, perhitungan statistik, dan pengubahan skala data. Hal ini dilakukan untuk menyesuaikan jenis data dan kebutuhan saat menganalisis data [25].
4. Analisis data merupakan proses untuk mendapatkan sebuah informasi atau wawasan yang dibutuhkan. Menganalisis data sering kali menggunakan metode statistik, teknik data *mining*, *machine learning*, dan lain-lain. Tujuan adanya analisis data untuuk mendapatkan sebuah pola, tren, dan hubungan dalam data-data yang dikumpulkan.
5. Interpretasi dan visualisasi data ialah proses lanjutan setelah menganalisis data. Hasil analisis data dipresentasikan melalui visualisasi agar mudah dipahami dan sebagai sarana informatif. Visualisasi data yang dilakukan bisa berupa grafik, diagram, atau peta untuk membantu memahami pola atau tren yang ada sehingga dalam proses komunikasi hasil analisis kepada pemangku kepentingan mudah dicerna.
6. Penyimpanan dan pengelolaan data merupakan proses akhir yang dilakukan pada pengolahan data. Pada proses ini data disimpan dalam format yang aman dan terorganisir. Penyimpanan data melibatkan penggunaan basis data dan sistem menajemen data serta penyimpanan *Cloud* guna memastikan aksesibilitas data, keamanan, dan keberlanjutan data sehingga data masih bisa diakses dalam waktu jangka panjang.

Data juga dapat diolah untuk menggambarkan perubahan bentuk data menjadi informasi. Pengolahan data atau basis data mampu mendefinisikan

kumpulan data yang saling terhubung, disimpan tanpa harus khawatir terdapat data berulang redundansi sehingga dapat memenuhi berbagai kebutuhan. Data yang sudah diolah menjadi penyedia informasi yang mencakup pengumpulan, proses pengolahan data, dan pengawasan hasil data olahan.

2.4 Bahasa Pemprograman Pyhton

Bahasa pemprograman menjadi jembatan untuk pengaplikasian *machine learning* salah satunya ialah bahasa pemrograman python. Python menjadi bahasa pemprograman yang memiliki pendekatan yang fokus pada kesederhanaan dan kejelasan sintaksis [26]. Sintaksis yang bersih dan terstruktur sehingga pengembang mampu menghasilkan kode yang mudah dibaca dan dipelihara seperti penggunaan indentasi sebagai metode menunjukkan blok kode. Python menawarkan beragam pustaka dan modul yang dapat diakses dengan mudah oleh pengembang.

Pada bidang analisis data dan kecerdasan buatan penggunaan Python sangat populer dilengkapi dengan pustaka-pustaka yang disediakan seperti *NumPy*, *Pandas*, *matplotlib*, *Tensorflow*, hingga *PyTorch*. Selain itu, Python juga memberikan *tools* yang membantu pengembang dalam memanipulasi dan menganalisis data, serta memudahkan dalam membangun model pembelajaran mesin.

Tensorflow merupakan pustaka pembelajaran mesin atau *machine learning* yang populer. *Tensorflow* sering digunakan karena mudah dikembangkan dan memiliki komputasi tingkat tinggi. Selain itu, *tensorflow* memberikan dukungan untuk berbagai jenis model pembelajaran termasuk jaringan saraf konvolusi (CNN) dan jaringan saraf *rekurent* (RNN), serta mencakup algoritma populer seperti regresi linear, klasifikasi, dan pemodelan bahasa alami.

2.5 *Deep Learning*

Pembelajaran mendalam yang sering disebut *deep learning* merupakan bagian dari bidang penelitian *machine learning* dan kecerdasan buatan dimana proses komputasi dilakukan melalui beberapa lapisan proses guna mempelajari representasi data diberbagai tingkat. *Deep learning* terfokus pada pengembangan

dan penerapan jaringan saraf (*neural networks*). Pada dasarnya jaringan saraf terdiri dari banyak lapisan (*layer*) yang saling terhubung [27]. Dengan kemampuan ekstrasi fitur yang relevan dan abstrak dari data secara otomatis membuat *deep learning* digemari. Hal ini terjadi karena tidak lagi menentukan fitur secara manual.

Pada penelitian ini penggunaan *deep learning* dalam metode *matchmaking* dengan memanfaatkan algoritma *machine learning* untuk memodelkan pola dan hubungan kompleks dalam data. *Deep learning* mampu memprediksi pasangan yang cocok berdasarkan data yang diinputkan. Data yang dikumpulkan berupa preferensi, perilaku pengguna, dan minat. Selain itu, *deep learning* mampu menganalisis teks dengan pembuatan model yang mampu memahami dan mengekstrak informasi seperti pesan, ulasan, hingga bio pengguna. Sehingga, pemahaman yang mendalam tentang pengguna mampu menghasilkan rekomendasi yang relevan [28].

2.6 *Natural Languange Processing* (NLP)

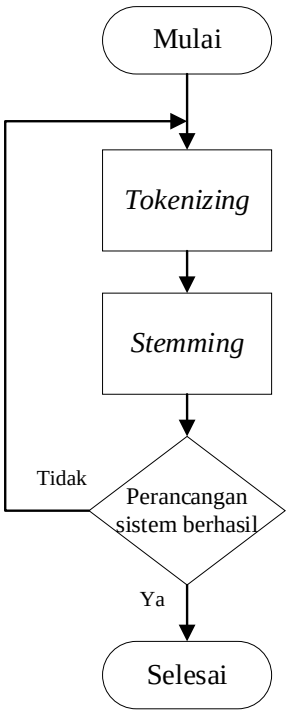
Natural Languange Processing termasuk kedalam domain *deep learning* yang merupakan cabang ilmu komputer dan kecerdasan buatan yang terfokus pada pemahaman, analisis, dan bahasa. Tujuan penggunaan NLP untuk membangun komunikasi dan interaksi yang baik antara manusia dan mesin melalui bahasa manusia. Pada NLP mesin mulai mempelajari untuk memahami, dan menganalisis, serta menghasilkan hasil teks dalam bahasa yang dapat dipahami manusia.

NLP menggunakan beberapa teknik untuk memproses dan memahami bahasa manusia. Teknik tersebut meliputi sebagai berikut:

1. *Tokenization* atau tokenisasi merupakan proses membagi teks menjadi kata per kata atau token. Tokenisasi merupakan langkah pertama dalam proses NLP dan mempengaruhi proses selanjutnya. Proses membagi teks menjadi kata atau token dijadikan sebagai *input* untuk algoritma NLP. Tokenisasi dibagi menjadi beberapa cara yaitu, *word tokenization*, *character tokenization*, dan *subword tokenization*. Tokenisasi memiliki keunggulan seperti mengurangi kompleksitas dan mengurangi memori. Namun,

tokenisasi membuat ambiguitas, sehingga tokenisasi mengalami kesulitan dalam mengidentifikasi kata yang benar [29].

2. *Stemming* merupakan proses transformasi kata ke bentuk kata dasar atau *root word*. Pada proses ini terjadi penghapusan imbuhan dan akhiran kata, sehingga kata-kata yang memiliki akar kata yang sama diidentifikasikan sebagai satu kata yang sama [28]. *Stemming* memudahkan dalam proses pencarian dan analisis karena adanya pengurangan variasi kata.
3. *Lemmatization* bertujuan untuk mentransformasikan kata dalam bentuk kata dasar atau *root word*. Namun, tidak seperti *stemming*. *Lemmatization* mempertimbangkan konteks dan aturan tata bahasa yang berlaku sehingga hasil yang diberikan lebih akurat [30].
4. *Part of Speech* (POS) *tagging* merupakan proses yang ada dalam NLP dimana POS *tagging* mengkategorikan setiap kata dalam sebuah kalimat atau dokumen berdasarkan dengan kategorik gramatiknya, seperti *noun, verb, adverb, adjective*, dan lain-lain. POS *tagging* mampu memudahkan proses pemahaman struktur dan arti kalimat, serta pada proses pencarian dan analisis. POS *tagging* dapat dilakukan dengan berbagai cara salah satunya dengan *rule-based tagging* dan *stochastic tagging* [31].
5. *Named Entity Recognition* (NER) mirip seperti POS *tagging* yang membedakan hanyalah jika NER menambahkan kategori sesuai gramatiknya, seperti nama orang, nama tempat, nama organisasi, tanggal, waktu, dan lain-lain [32].
6. *Sentiment analysis* bertujuan untuk menentukan data yang diberikan bersifat positif, negatif, atau netral. *Sentiment analysis* mampu memahami opini dan perasaan pelanggan, serta memudahkan dalam proses pencarian dan analisis. *Sentiment analysis* memiliki beberapa metode, seperti *rule-based* dan *machine learning*. Pada *rule-based* melibatkan penggunaan aturan tertentu untuk mengklasifikasikan kata. Sedangkan metode *machine learning sentiment analysis* melibatkan algoritma klasifikasi kata ke dalam sentimen yang sesuai [33]. Diagram alir NLP dapat dilihat pada Gambar 2.3 sebagai berikut.



Gambar 2.3 Diagram Alir NLP

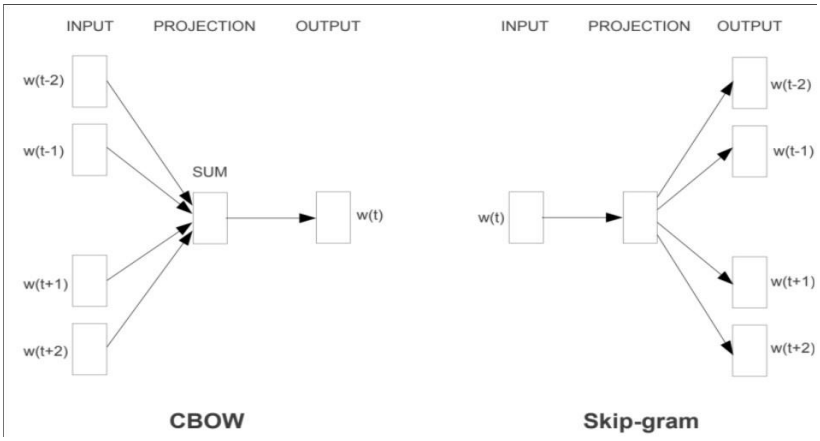
Pada Gambar 2.3 merupakan proses penggunaan NLP secara general atau umum. Namun, proses NLP yang sebenarnya bisa jauh lebih kompleks dan bervariasi tergantung pada tugas dan tujuan model yang dibuat. Akan tetapi, langkah dan proses yang sudah dijelaskan di atas merupakan proses yang umumya terdapat pada proses pengolahn teks [34]. NLP dapat digunakan dalam berbagai aplikasi, seperti aplikasi pencari, sistem pengolahan teks, *chatbot*, dan lain-lain.

2.7 *Word Embedding*

Word embedding merupakan representasi vektor yang digunakan untuk mewakili kata-kata dalam *natural language processing* atau NLP. Pada dasarnya *word embedding* digunakan untuk mengubah kata-kata menjadi bentuk numerik yang dapat dipahami oleh model pembelajaran mesin. Dalam *word embedding* setiap kata diwakili oleh sebuah vektor numerik multidimensi. Hal ini terjadi karena setiap dimensi dalam vektor tersebut merepresentasikan suatu fitur atau aspek tertentu. Proses representasi ini memungkinkan komputer dapat memahami hubungan kata berdasarkan dengan kedekatan atau kesamaan dalam ruang vektor.

Salah satu teknik *word embedding* yaitu Word2Vec. Word2Vec menggunakan model pembelajaran mesin berbasis jaringan saraf tiruan untuk mempelajari representasi vektor kata. Terdapat dua pendekatan dalam Word2Vec yaitu *Continuous Bag-of-Words* (CBOW) dan *Skip-gram*. Pada Word2Vec terdiri dari sebuah *hidden layer* yang disebut dengan *projection layer*. *Neural network* dilatih menggunakan *stochastic gradient descent* dengan menggunakan algoritma *backpropagation*. *Projection layer* berfungsi sebagai pemetaan dari kata-kata dalam konteks n-gram ke dalam bentuk vektor kontinu. Kata-kata dengan intensitas yang tinggi dalam konteks n-gram cinderung memiliki bobot yang sama, sehingga terdapat korelasi antara katakata tersebut.

Bobot atau *weights* menghubungkan lapisan masuk (*input layer*) dengan lapisan tersembunyi (*hidden layer*), dan lapisan tersembunyi (*hidden layer*) dengan lapisan keluaran (*output layer*). *Weights* diantara *input layer* dengan *hidden layer* diwakili dengan matriks W dengan ukuran $V \times N$, dimana V merupakan dimensi *input layer* dan N merupakan dimensi *hidden layer*. Selain itu *weights* antar *hidden layer* dengan *output layers* diwakili dengan matriks W dengan ukuran $N \times V$ [35]. Arsitektur Word2Vec dapat dilihat pada Gambar 2.4 berikut.

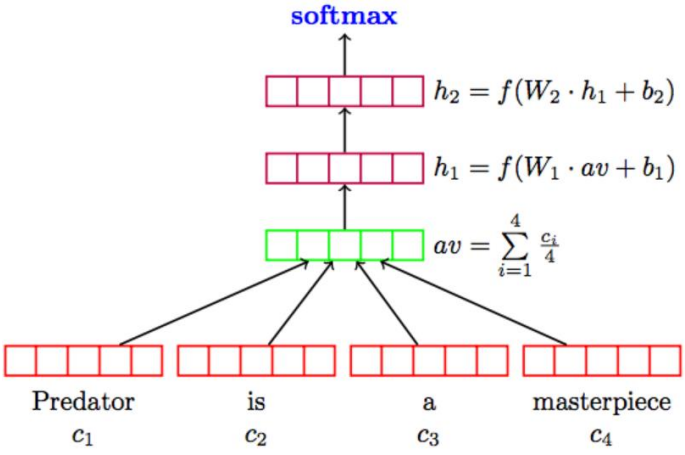


Gambar 2.4 Arsitektur Word2Vec

Berdasarkan Gambar 2.4 model CBOW menggunakan kata-kata yang berada di sebelah kiri dan kanan kata target yang memiliki sebuah batasan yang diperoleh jendela (*window*) untuk memprediksi kata target. Sedangkan dalam

model *skip-gram* sebuah kata digunakan untuk memprediksi kata-kata yang beada di sebelah kiri dan kanan kata tersebut dengan batasan yang sama diperoleh jendala (*window*). Setiap kata yang digunakan merupakan sebuah inputan yang diubah menjadi vektor *one-hot*. Perbedaan utama antar dua buah pendekatan ini terletak pada cara memprediksi kata. Pada model CBOW terdapat lapisan perantara (*intermediate layer*) untuk menghitung rata-rata atau *mean* dari vektor kata *input* karena CBOW menerima sejumlah *n* kata sebagai *input* [36].

Salah satu teknik *word embedding* lainnya yaitu *Deep Averaging Network* (DAN) merupakan arsitektur jaringan saraf yang memanfaatkan NLP. Arsitektur DAN dirancang untuk menyelesaikan tugas klasifikasi teks, analisis, dan pemodelan urutan teks. Pada arsitektur DAN mengambil rata-rata vektor representasi kata dalam sebuah kalimat yang diteruskan menggunakan lapisan jaringan saraf untuk memperoleh representasi kalimat. DAN mampu menyelesaikan masalah pemprosesan bahasa alami dengan jumlah data yang besar dan tidak memperlukan struktur tata bahasa serta urutan kata.

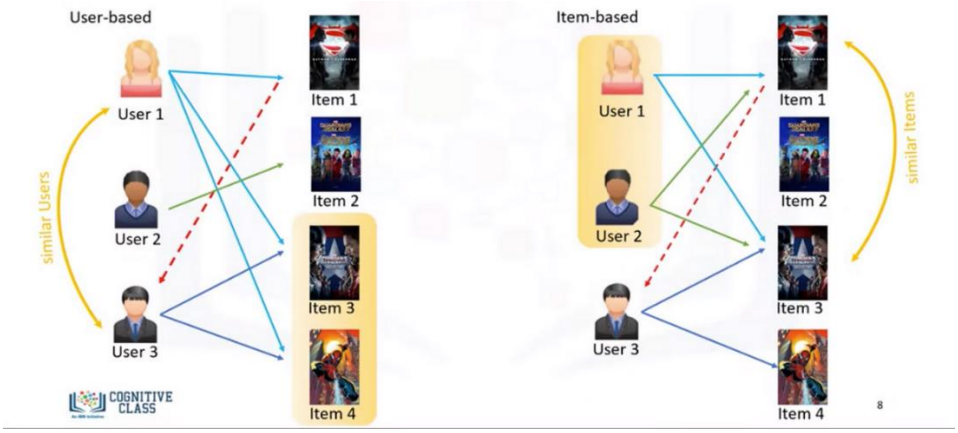


Gambar 2.5 Arsitektur *Deep Averaging Network*

Pada Gambar 2.5 menjelaskan tentang arsitektur DAN dibuat dari rata-rata vektor *embedding* yang diteruskan melalui lapisan tersembunyi. Pada lapisan tersembunyi menghitung transformasi *linear* dan fungsi aktivasi *non-linear*. Selanjutnya, terdapat lapisan *output* yang bertugas untuk menyelesaikan tugas yang diberikan seperti klasifikasi teks, pengurut teks, dan lain-lain.

2.8 *Collaborative Filtering*

Collaborative filtering merupakan sebuah metode yang digunakan dalam sistem rekomendasi untuk menghasilkan rekomendasi personel berdasarkan prefrensi pengguna. Metode ini terfokus pada pengumpulan informasi dari pengguna yang memiliki preferensi serupa guna menghasilkan sebuah rekomendasi kepada pengguna lainnya yang memiliki kesamaan preferensi. Metode ini tidak membutuhkan informasi eksplisit tentang *item* yang direkomendasikan, namun mengandalkan data historis yang dikumpulkan dari interaksi pengguna dengan *item* untuk mengidentifikasi sebuah pola yang dihasilkan dari kesamaan prefernsi [36]. Perbedaan *user-based* dan *item-based* dapat digambarkan melalui gambar berikut.



Gambar 2.6 *Collaborative Filtering*

Pada Gambar 2.6 *collaborative filtering* memiliki dua pendekatan yaitu *user-based* dan *item-based*. Dalam pendekatan *user-based* kesamaan antar pengguna dihitung berdasarkan riwayat interaksi mereka terhadap sebuah *item* yang sama. Pengguna mendapatkan preferensi yang sama dan kemudian memberikan rekomendasi pada pengguna lainnya. Pendekatan *item-based* menghitung kesamaan antar *item* berdasarkan pola interaksi pengguna dengan *item* tersebut. *Item* ini memiliki pola interaksi yang serupa yang dilanjutkan dengan membuat rekomendasi.

Mampu mengatasi kekurangan informasi dan keberagaman prefrensi yang menjadikan *collaborative filtering* sering digunakan. Selain itu, *collaborative*

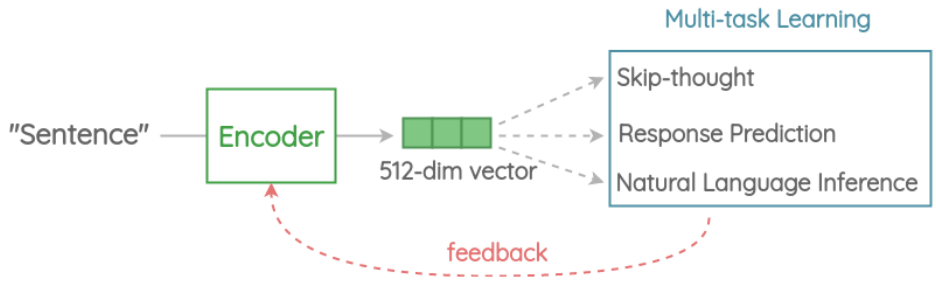
mampu menangani skala besar data dan dapat menghasilkan rekomendasi yang personal dan relevan. Namun, terdapat beberapa tantangan yaitu *cold-start problem* ketika terdapat pengguna baru atau *item* baru yang belum memiliki data historis [36].

2.9 *Universal Sentence Encoder* (USE)

Universal Sentence Encoder atau USE merupakan sebuah model yang dapat digunakan untuk mengkodekan sebuah kalimat ke dalam vektor *embedding* yang berguna untuk memproses bahasa alami (NLP). Model yang menggunakan USE mempelajari makna dan sifat semantik dari kalimat dari data yang sangat besar. Model USE dirancang guna memaksimalkan kinerja *transfer learning* ke tugas-tugas NLP [37].

Transfer learning ialah model yang sudah dilatih digunakan untuk menyelesaikan tugas lainnya tanpa harus dilatih kembali. Pada USE *transfer learning* digunakan pada proses pengkodean kalimat ke dalam vektor *embbeding* yang langsung melakukan tugasnya sebagai klasifikasi teks. Pada model USE *transfer learning* melibatkan dua tahap yaitu model dilatih pada tugas utama yang sangat luas atau general. Hal ini dilakukan untuk proses pemahaman model USE dalam memaknai sebuah struktur kalimat dalam bahasa yang beragam dan menghasilkan representasi vektor yang padat [3].

Penggunaan vektor *embedding* pada model USE merupakan representasi numerik dari sebuah kalimat yang merepresentasikan makna dari kalimat tersebut. Model USE menggunakan vektor *embedding* yang memiliki dimensi 512 yang dihasilkan melalui proses *deep learning* [37]. Proses *deep learning* yang dilakukan memanfaatkan *transformer*/*deep learning network* (DAN) untuk menghasilkan *output* berdimensi 512 yang dipasangkan dengan kalimat. Proses vektor *embedding* dapat dilihat pada Gambar 2.7 berikut.



Gambar 2.7 Vektor *Embedding*

Pada Gambar 2.7 menjelaskan dalam merancang *encoder* yang menggambarkan setiap kalimat dilakukan penyematan kalimat yang berdimensi 512. Penyematan tersebut digunakan untuk menyelesaikan berbagai tugas dan berdasarkan *error* atau kesalahan yang terjadi penyematan segera diperbarui. Hal ini terjadi karena penyematan harus digunakan pada tugas umum, sehingga fitur-fitur yang paling informatif diambil dan menghilangkan *noise*. Pada tahap selanjutnya ialah model USE yang sudah dipelihara disesuaikan dengan tugas yang diberikan seperti, melakukan klasfikasi teks. Pada proses ini model USE beradaptasi dengan vektor yang diperoleh pada tahap sebelumnya sesuai dengan tugas yang diberikan.

2.10 *Tensorflow Recommendation*

Tensorflow Recommendation atau yang biasa disebut TFRS merupakan sebuah pustaka atau *library* untuk membantu sistem rekomendasi menggunakan *Tensorflow*. Penggunaan TFRS dirancang khusus untuk memudahkan dalam proses membangun sebuah model rekomendasi. TFRS menyediakan komponen yang membantu sistem rekomendasi, seperti pemprosesan data, pembuatan fitur, pemodelan, hingga evaluasi [38]. Dengan menggunakan TFRS dapat menggabungkan fitur-fitur yang kuat seperti pemprosesan grafik, jaringan saraf, dan optimisasi yang kuat sehingga menghasilkan model rekomendasi yang akurat.

Sistem rekomendasi merupakan salah satu aplikasi yang mengumpulkan data interaksi pengguna dan menggunakannya untuk membantu *user* atau pengguna

menemukan informasi yang diinginkan. Cara kerja TFRS terdapat beberapa langkah yang dijelaskan sebagai berikut.

1. *Data Preparation* atau persiapan data melibatkan beberapa langkah yaitu, data *loading*, data *preprocessing*, data *splitting*, data *encoding*, dan data *caching*. Data *loading* merupakan proses memuat data interaksi pengguna. Dilanjutkan dengan data *preprocessing* yaitu melakukan pembersihan data, menghapus data yang tidak relevan, dan mengubah ke format yang diinginkan. Lalu, data *splitting* yaitu membagi dataset menjadi data pelatihan dan data validasi. Kemudian, data *encoding* merupakan data yang masuk ke vektor *embedding*. Proses terakhir yaitu data *caching* merupakan proses penyimanan data yang sudah diolah ke dalam *cache* untuk mempercepat oleh akses model. Data *preparation* digunakan untuk memastikan kualitas data yang digunakan dalam membangun sistem rekomendasi. Hal ini dilakukan melihat data yang baik dan terstruktur dapat membantu meningkatkan kinerja model dan akurasi yang dihasilkan [39].
2. *Model formulation* merupakan langkah pembuatan model yang merepresentasikan data pengguna. Pada *model formulation* melibatkan proses *feature engineering* yaitu pembuatan fitur dari data yang sudah diolah menggunakan teknik *embedding*. Selanjutnya, *model selection* yaitu memilih model mana yang lebih relevan dan sesuai dengan tugas yang diberikan. lalu, model di-*trainning* atau dilatih untuk membantu mengidentifikasikan tugas yang ada. Pada tahap akhir, model dievaluasi atau diperiksa kinerja model dan memperbaiki jika terdapat *error* atau kesalahan [39].
3. *Training* pada TFRS ialah salah satu proses membangun sistem rekomendasi yang dimana model yang sudah dibuat melalui proses pelatihan untuk membantu mengidentifikasikan tugas yang diberikan [39].
4. *Evaluation* atau penilaian kinerja model rekomendasi yang sudah dilatih. Pada umunya proses evaluasi melibatkan metrik evaluasi seperti *recall, precision, f-1 scrore,* dan lain-lain. Metrik evaluasi mengevaluasi seberapa baik kinerja model pada sistem rekomendasi. Selain itu, validasi model

dilakukan untuk memastikan hasil rekomendasi yang diberikan sesuai dan relevan.

5. *Deployment* merupakan penerapan pada model rekomendasi yang sudah dibangun ke dalam sistem atau aplikasi. Pada proses ini juga terdapat pengujian model yang sudah dihubungkan ke sistem atau aplikasi untuk memastikan model sudah berjalan dengan baik dan berhasil memberikan hasil rekomendasi yang akurat.

Selain itu, TFRS memiliki beberapa fitur yaitu *Tensorflow Recommendation Addons* yang bertujuan untuk proyek komunitas menggunakan *embedding* untuk mempertahankan privasi pengguna. *Tensorflow Lite* yaitu menjadi sebuah solusi proses bahasa alami dengan letensi rendah dan kinerja yang bagus tanpa harus mengirim data pengguna ke *server*. *Tensorflow Federated* yaitu sebuah *framework* untuk pembelajaran komputasi. Selanjutnya yaitu fitur *Tensorflow Rangking* merupakan sebuah pustaka untuk membangun model *learning to rank* dengan skala dan *neural* [40].

2.11 Kajian Pustaka

Penelitian ini berlandaskan dari penelitian-penelitian terdahulu, baik dari landasan teori, metode, atau teknik penelitian yang digunakan, maupun jenis penelitiannya yang berkaitan dengan topik penelitian yaitu mengenai pembuatan model *machine learning* menggunakan metode USE dan TFRS untuk menghasilkan sistem rekomendasi. Berikut ini adalah beberapa penelitian yang menjadi landasan dari penelitian yang dilakukan.

Pada penelitian sebelumnya telah membuktikan bahwa klasifikasi atau prediksi memperoleh hasil identifikasi yang akurat dan tepat. Hasil yang diperoleh pada penelitian ini menunjukan akurasi sebesar 69% pendekatan yang digunakan lebih unggul dibanding menggunakan metode dasar (*baseline*) [41]. Pada penelitian ini menggunakan *question sentence embedding* untuk melakukan klasifikasi. Pendekatan yang ditawarkan yaitu menyederhanakan tahap ekstraksi fitur dengan tidak mengekstraksi fitur-fitur seperti entitas bernama yang ada dalam pertanyaan yang jumlahnya sedikit karena panjang pertanyaan yang pendek dan juga fitur-fitur

seperti hipernim dan hiponim dari sebuah kata yang membutuhkan ekstensi *WordNet* dan membuat sistem lebih tergantung pada sumber eksternal. Pendekatan yang diusulkan menganjurkan penggunaan *universal sentence embedding* dengan *transformer encoder* untuk mendapatkan vektor *embedding* pada *level* kalimat dengan ukuran tetap, dan kemudian menghitung kesamaan semantik antara vektor-vektor tersebut untuk mengklasifikasikan pertanyaan sesuai dengan kategori-kategori yang sudah ditentukan [41].

Sebuah penelitian telah dilakukan untuk menggunakan *universal sentence encoder* dalam mengklasifikasikan sentimen dari *tweet-tweet* yang berkaitan dengan COVID-19. Model ini berhasil mencapai tingkat akurasi sebesar 78,062%, yang lebih baik dibandingkan dengan klasifikasi mesin tradisional [4]. Selama masa pandemi COVID-19, individu dan organisasi media menggunakan media sosial untuk menulis dan mendokumentasikan status kesehatan mereka serta berita terkini seputar *coronavirus*. Dengan menggunakan *tweet-tweet* (sentimen) ini yang berkaitan dengan *coronavirus* dan menganalisanya menggunakan model komputasi, dapat membantu pengambil keputusan dalam mengukur opini publik dan menghasilkan temuan yang menarik. *Dataset* yang digunakan dalam penelitian ini dikumpulkan dari Twitter dan diklasifikasikan menjadi tiga kategori, yaitu positif, netral, dan negatif. Model *embedding* kalimat ini bertujuan untuk menentukan makna urutan kata-kata daripada kata-kata individu [4]. Selain itu, model ini membagi *dataset* menjadi data latih dan data uji, serta bergantung pada kesamaan kalimat untuk mendeteksi kelas sentimen. Dengan menggunakan model analisis sentimen berbasis *deep learning* yang menggunakan *universal sentence encoder* dapat membantu pengambil keputusan dalam mengukur opini publik dan menghasilkan temuan yang menarik seputar *coronavirus*.

Pada penelitian selanjutnya, membahas tentang sistem rekomendasi yang menawarkan cara lebih efisien dalam mencari barang yang mereka inginkan. Penelitian ini membangun sistem rekomendasi menggunakan metode *content-based filtering* dan *Term Frequency Inverse Document Frequency* (TF-IDF). Pada sistem rekomendasi yang dibangun menyediakan solusi meningkatkan kesadaran merek pelanggan dan meminimalisir kegagalan transaksi akibat adanya informasi

yang kurang lengkap baik *offline* maupun *online*. Dataset yang dipakai meliputi 258 kode produk dengan 8 kategori dan 33 kata kunci. Hasil perhitungan TF-IDF menunjukkan nilai bobot sebesar 13.854 saat menampilkan rekomendasi produk terbaik dan memiliki nilai akurasi sebesar 96,5% dalam memberikan rekomendasi [42].

Penelitian selanjutnya membahas tentang sistem *matchmaking* dalam permainan *multiplayer* berguna dalam komunikasi interaktif antar pemain secara bersamaan. Faktor yang mempengaruhi *matchmaking* ialah *rating* atau *matchmaking rating* (MMR). Nilai MMR didapatkan dari standarisasi yang dilakukan secara manual oleh pemain. Selain itu, MMR tidak menjamin keseimbangan dalam permainan. Oleh karena itu, penggunaan *recommender system* sangat diperlukan untuk memberikan preferensi sesuai dengan kebutuhan pengguna. Pada penelitian ini menggunakan metode *hybrid recommender system* yang menggabungkan *content-base* dan *user-based collaborative filtering*. *Dataset* yang digunakan meliputi 9 orang dalam 204 pertandingan. Pemain yang direkomendasikan adalah pemain yang memiliki profil pengguna yang sesuai. Hasil performa sistem yang sudah diuji memalui matriks evaluasi meliputi nilai *recall*, *precision*, dan *f-measure*. Hasil rekomendasi *matchmaking* mencapai 77,7% untuk nilai *recall*, 77,7% untuk nilai *precision*, dan 77,7% untuk nilai *f-measure* dengan 20 rekomendasi *matchmaking* [43].

Hasil penelitian yang memanfaatkan perangkat *Internet of Things* (IoT) dan menggunakan model *Artificial Intelligence Agriculture Recommendation Model* (AIARM) untuk meningkatkan produktivitas pertanian hingga peniliaian lahan pertanian. Metode ini menggabungkan jaringan sensor dan program kecerdasan buatan seperti jaringan saraf dan penerapan *multi-layer perceptron* guna menentukan kesiapan tanaman, prediksi tanaman, dan rekomendasi pemupukan. Pada model ini menempatkan beberapa kategori keputusan yaitu layak, sesuai, cukup adil, dan tidak sesuai. Dengan adanya preferensi pengambilan keputusan mampu membantu petani dengan akurasi yang lebih tinggi. Pada penelitian ini mendapatkan nilai akurasi sebesar 74% untuk *neural network* dan 76% untuk *muti-*

layer perceptron. Ini menunjukkan bahwa otomatisasi pertanian menunjukkan nilai yang signifikan dalam pengembangan sistem rekomendasi [44].

Berdasarkan beberapa hasil penelitian sebelumnya yang menjadi acuan dalam kajian pustaka, maka dilakukan pengintegrasian permasalahan yang telah diidentifikasi dan mencoba untuk menyelesaikan dengan menggunakan metode yang telah dijelaskan dalam penelitian tersebut. Penelitian yang dilakukan ini menggunakan metode USE dan TFRS untuk mendapatkan hasil rekomendasi *user* yang sesuai dengan *background* pengguna dengan menambahkan dua buah parameter yaitu *skills* dan *job interst*. Dari hasil rujukan sebelumnya terdapat kekurangan yang perlu diatasi. Kekurangan tersebut meliputi masalah *cold-start*, kurangnya data, akurasi prediksi yang rendah, masalah privasi, dan kompleksitas integrasi model pembelajaran. Selain itu, terdapat pula keterbatasan dalam hal interpretabilitas.

Oleh karena itu, diperlukan algoritma yang efektif untuk digunakan dalam sistem rekomendasi. Algoritma *Universal Sentence Encoder* (USE) dan *Tensorflow Recommendation* (TFRS) telah terbukti menjadi solusi terbaik untuk mengatasi masalah saat ini. Kelebihan dari algoritma ini adalah implementasinya yang relatif mudah digunakan tanpa memerlukan pemahaman yang mendalam terhadap banyak parameter. Hal ini memungkinkan penanganan kendala seperti *overfitting* atau *underfitting* tanpa memerlukan banyak pengaturan parameter yang rumit.