BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam penelitian ini akan digunakan enam tinjauan studi yang nantinya dapat mendukung penelitian. Berikut ini merupakan tinjauan studi yang digunakan dalam penelitian, dapat dilihat pada Tabel 2.1 Daftar Literatur.

Tabel 2.1 Daftar Literatur

No.	Penulis	Judul	Metode dan Hasil
01	Omar Hussein, Hachem Sfar, Jelena Mitrovic, Michael Granitzer (2020)	*NLP_Passau at SemEval-2020 Task 12: Multilingual Neural network for Offensive Language Detection in English, Danish and Turkish*	Algoritma CNN dan RNN dengan hasil akurasi Turki 83.34%, Denmark 88.89%, Inggris 77.87%; presisi Turki 74%, Denmark 88%, Inggris 75%; dan recall Turki 65%, Denmark 59%, Inggris 75%
02	Crisanadenta Wintang Kencana, Erwin Budi Setiawan, Isman Kurniawan (2020)	*Hoax Detection on Twitter using Feed-forward and Back-Propagation Neural networks Method*	Algoritma BNN dengan hasil akurasi 78.76%, presisi 73.40% dan recall 75%
03	Theresia Hendrawati, Christina Purnama Yanti (2021)	*Analysis of Twitter Users Sentiment against the Covid-19 Outbreak Using the Backpropagation Method with Adam*	Algoritma BNN dengan Adam Optimization dengan akurasi 70%, presisi 65% dan recall 67 %

No.	Penulis	Judul	Metode dan Hasil
		Optimization	
04	Euis Saraswati, Yuyun Umaidah, Apriade Voutama (2021)	Penerapan Algoritma *Artificial Neural network* untuk Klasifikasi Opini Publik Terhadap Covid-19	Algoritma BNN dengan hasil akurasi 88.62%, presisi 91.5% dan recall 95.73%
05	Mhd. Denry Aruna Nasution, Jaya Tata Hardinata, Irfan Sudahri Damanik (2019)	Jaringan Syaraf Tiruan *Backpropagation* untuk Klasifikasi Data Tilang Berdasarkan Jenis Pelanggaran	Algoritma BNN dengan hasil akurasi 95%, presisi 87%, dan recall 93%
06	Conor O'Sullivan (2020)	*Deep* Neural network *Language Identification*	Algoritma DNN dan n-gram karakter dengan hasil akurasi 98.26%, presisi 95% dan recall 97%

2.1.1 Literatur 1

Makalah ini menjelaskan model Neural network (NN) yang digunakan untuk berpartisipasi dalam *OffensEval*, Tugas 12 dari lokakarya *SemEval* 2020. Tujuan dari tugas ini adalah untuk mengidentifikasi pidato ofensif di media sosial, khususnya dalam tweet. Model yang kami gunakan, *C-BiGRU*, terdiri dari *Convolutional* Neural network *(CNN)* bersama dengan *Bidirectional Recurrent* Neural network *(RNN)*. Representasi numerik multi dimensi (*embedding*) untuk setiap kata dalam tweet, yaitu digunakan oleh model, ditentukan menggunakan *fastText*. Ini digunakan dengan kumpulan data berlabel tweet untuk melatih model dalam mendeteksi kombinasi kata yang dapat menyampaikan serangkaian arti.

Dalam makalah ini, sistem yang kami gunakan untuk berpartisipasi dalam *OffensEval* 2020 ditampilkan dan penjelasan rinci diberikan tentang arsitektur model *C-BiGRU* dan *pre-processing* data yang digunakan. Adapun *dataset* untuk pelatihan yang digunakan adalah 14.100 untuk bahasa Inggris, 2.961 untuk bahasa Denmark, dan 31.277 bahasa Turki. Model ini mencapai skor F1 rata-rata makro sebesar 90,882%, 76,76%, dan 76,70% untuk bahasa Inggris, Turki, dan Denmark masing-masing pada pelabelan *dataset OffensEval 2*.

2.1.2 Literatur 2

Media sosial adalah salah satu cara untuk menghubungkan setiap individu di dunia. Itu juga digunakan oleh orang-orang yang tidak bertanggung jawab untuk menyebarkan *hoax*. *Hoax* adalah berita bohong yang dibuat seolah-olah benar. Hal ini dapat menimbulkan kecemasan dan kepanikan di masyarakat. Hal ini dapat mempengaruhi kondisi sosial dan politik. Pada era saat ini, media sosial yang paling populer adalah Twitter. Ini adalah tempat untuk berbagi informasi dan pengguna di seluruh dunia dapat berbagi dan menerima berita dalam pesan singkat atau disebut tweet. Deteksi *hoax* mendapatkan minat yang signifikan dalam dekade terakhir.

Metode pendeteksian hoax yang ada saat ini berbasis konten berita atau konteks sosial dengan menggunakan fitur berbasis pengguna. Dalam studi ini, kami menghadirkan deteksi *hoax* berdasarkan jaringan saraf *Feed Forward & Back Propagation*. Dalam pengembangannya, kami menggunakan dua metode *vektorisasi, TF-IDF* dan *Word2Vec*. Model kami dirancang untuk mempelajari fitur klasifikasi berita hoax secara otomatis melalui beberapa *hidden layer* dibangun ke dalam jaringan saraf. Jaringan syaraf sebenarnya menggunakan

kemampuan otak manusia yang mampu memberikan rangsangan, proses, dan *output*. Ini dikerjakan oleh *neuron* untuk memproses setiap informasi yang masuk, kemudian diproses melalui koneksi jaringan, dan akan terus belajar untuk menghasilkan kemampuan melakukan klasifikasi.

Model yang kami usulkan akan membantu untuk memberikan solusi yang lebih baik untuk mendeteksi *hoax*. Pengumpulan data diperoleh melalui *crawling* menggunakan Twitter API dan mengambil data sesuai dengan kata kunci dan tagar dari tanggal 13 sampai 16 Maret 2020. Didapatkan 50.646 tweet yang mana sebanayak 25.021 *class hoax* dan 25.624 *class non-hoax*. Dari *dataset* tersebut dibagi menjadi 80% data *train* dan 20% data *test*. Jaringan saraf akurasi tertinggi diperoleh menggunakan *TF-IDF* dengan hasil 78,76% dibandingkan menggunakan *Word2Vec* dengan hasil 67,38%. Kami juga menemukan bahwa kualitas data memengaruhi kinerja.

2.1.3 Literatur 3

Penelitian ini memanfaatkan data dari Twitter dengan menganalisis tweet berbahasa Indonesia yang membahas tentang wabah virus Covid-19 untuk mengetahui pendapat pengguna Twitter tentang wabah virus Covid-19. Penelitian ini mencoba menganalisis sentimen untuk melihat opini pada pengguna tweet tentang virus Covid-19 dengan hasil sentimen Positif, Negatif atau Netral menggunakan *Multi-layer Perceptron* (MLP) dan *Backprogragation* dengan optimasi Adam. Pada penelitian ini mengumpulkan 200 dokumen (tweet) dalam bahasa Indonesia tentang Covid-19 yang di-tweet sejak November 2019 sampai dengan 30 Agustus 2020.

Setelah melalui 470 *epochs*, Model yang dihasilkan berhasil mendapatkan akurasi hingga 70% dengan skor f1 untuk kelas positif, negatif, dan netral masing-masing 0,77, 0,75, dan 0,5 dari nilai maksimum 1. Hal ini menunjukkan bahwa model pada penelitian ini cukup berhasil dalam melakukan proses klasifikasi sentimen untuk tweet berbahasa Indonesia dengan Tema Covid-19.

2.1.4 Literatur 4

Pada penelitian ini membuat sebuah sistem yang dapat melakukan klasifikasi opini publik terhadap Covid-19 menggunakan metode Backpropagation Neural network. Metode ini digunakan karena dapat melakukan klasifikasi data dengan jumlah fitur yang banyak dan beragam.

Penelitian ini berdasarkan data opini publik tentang virus yang telah beredar di banyak media sosial, termasuk media sosial Twitter. Di Twitter, ada beberapa diskusi terkait virus Covid-19. Perasaan positif atau negatif dapat ditemukan dalam opini di tweet. Sentimen sebuah tweet dapat diperhitungkan dan dievaluasi oleh pihak berwenang saat menangani virus Covid-19. Berdasarkan permasalahan tersebut, diperlukan klasifikasi analisis sentimen untuk mengetahui bagaimana virus Covid-19 dipersepsikan oleh masyarakat luas.

Algoritma Artificial Neural network (ANN) dan metode Backpropagation digunakan dalam penelitian ini. Pada pengujian tersebut, dihasilkan nilai presisi 91,5%, recall 95,73%, dan akurasi 88,62%. Hasilnya menunjukkan bahwa model Neural network sangat efektif untuk pengklasifikasian *text mining.*

2.1.5 Literatur 5

Bukti pelanggaran (tilang) diberikan kepada setiap orang yang melakukan aktivitas pelanggaran lalu lintas. Karena tingginya tingkat pelanggaran yang

terjadi menyebabkan data tilang yang tersimpan pada Kejaksaan Negeri Simalungun menjadi hambatan seperti kelebihan kapasitas ruang sidang. Dalam berkas tilang, ada beberapa poin penting yang menjelaskan tentang tilang secara mendetail seperti jenis barang bukti, kendaraan, pasal, kehadiran di persidangan, dan jenis operasi yang dilakukan. Klasifikasi dapat digunakan sebagai pilihan alternatif dalam pembagian jadwal sidang tilang. Klasifikasi tilang dalam penelitian ini menggunakan metode algoritma *Backpropagation*.

Susunan perhitungan *backpropagation* terdiri dari 2 (dua) bagian, yaitu bagian persiapan dan pegangan pengujian. Terdapat 100 unit informasi yang dipisahkan menjadi 2 (dua) yaitu 50 (lima puluh) unit informasi persiapan dan 50 (lima puluh) unit informasi uji. Informasi yang ada dimasukkan ke dalam desain jaringan klasifikasi yang terdiri dari input layer, hidden layer, output layer, learning rate dan tingkat errror. Adapun Tingkat akurasi terbaik yang diperoleh dari penelitian ini adalah sebesar 95.0%, dan presisi 87% dan recall 93%.

2.1.6 Literatur 6

Ada beberapa pendekatan berbeda untuk identifikasi bahasa dan, dalam artikel ini, kita akan membahasnya secara mendetail. Yaitu menggunakan Neural network dan karakter n-gram sebagai fitur. Pada akhirnya, penelitian ini menunjukkan bahwa akurasi lebih dari 98% dapat dicapai dengan pendekatan ini. Penelitian ini menggunakan 6.872.356 kalimat dalam 328 bahasa unik dengan pembagian 70% pelatihan, 20% validasi dan 10% untuk pengujian. Model yang digunakan adalah DNN yang memiliki 3 hidden layer dengan 250 node, dan 1 layer output untuk setiap bahasa dengan 6 node. Hingga menghasilkan tingkat akurasi sebesar 98.26%, presisi 95% dan recall 97%.

2.2 Artificial Neural network

Artificial Neural Network (ANN) adalah sebuah jaringan yang dirancang untuk menyerupai otak manusia yang bertujuan untuk melaksanakan suatu tugas tertentu (Haykin, 2009). Dalam pengimplentasian jaringan ini, biasanya menggunakan komponen elektronik atau disimulasikan pada aplikasi komputer.

Pada tahun 1943, Warren McCulloch dan Walter Pitts memperkenalkan konsep model Neural network untuk komputasi. Ini menandai langkah pertama di bidang ilmu jaringan Neural. Model mereka menggabungkan beberapa unit pemrosesan sederhana untuk meningkatkan daya komputasi secara keseluruhan.

ANN terdiri dari banyak *neuron* di dalamnya. *Neuron - neuron* ini akan dikelompokkan ke dalam beberapa *layer*. Ada tiga tipe *node* (*neuron*) yaitu, *input, hidden* dan *output*. Setiap relasi menghubungkan dua buah *node* dengan bobot tertentu dan juga terdapat arah yang menujukkan aliran data dalam proses (Kusrini dan Luthfi, 2009).

Setiap neuron yang ada di setiap lapisan Neural network terhubung dengan neuron di lapisan lainnya, kecuali layer input dan output. Hal ini hanya berlaku untuk lapisan-lapisan yang berada di antara keduanya. Informasi yang diterima oleh layer input akan diproses satu per satu oleh layer-layer dalam Neural network hingga mencapai layer output terakhir. Lapisan yang terletak di antara input dan output dikenal sebagai hidden layer. Namun, tidak semua Neural network memiliki hidden layer karena ada beberapa jenis yang hanya terdiri dari layer input dan output saja. Model ANN ditunjukkan dengan kemampuannya dalam emulasi, analisis, *prediksi*, dan *asosiasi*. Kemampuan yang dimiliki ANN dapat digunakan

untuk belajar dan menghasilkan aturan atau operasi dari beberapa contoh *input* yang dimasukan dan membuat prediksi tentang kemungkinan *output* yang akan muncul atau menyimpan karakteristik input yang diberikan kepada ANN.

2.3 Algoritma Backpropagation

Algoritma *Backpropagation* merupakan salah satu dari model jaringan saraf tiruan (JST). Pada tahun 1974 Paul Werbos pertama kali merumuskan algoritma ini dan digunakan Neural network yang dipopulerkan oleh Rumelhart bersama McClelland. Metode *Backpropagation* pada awalnya dirancang untuk neural network *feedforward*, tetapi pada perkembangannya, metode ini diadaptasi untuk pembelajaran pada model Neural network lainnya (Astuti, 2009). Model ini digunakan karena dalam kebanyakan kasus mengenali pola diperlukan mengidentifikasi ketika fakta-fakta atau fenomena akan terjadi lagi sebelum terjadi.

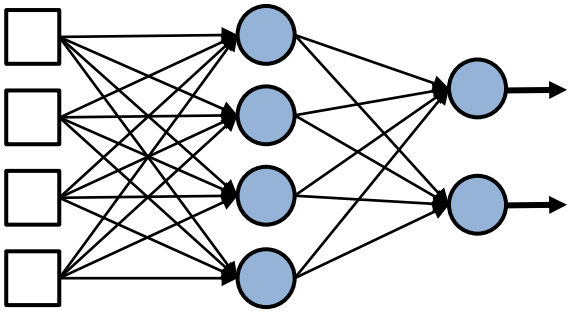
Pelatihan *Backpropagation* meliputi 3 fase:

i) Fase propagasi maju (*feedforward*) pola pelatihan masukan. Pola masukan dihitung maju mulai dari *layer* masukan hingga *layer* keluaran dengan fungsi aktivasi yang ditentukan,

ii) Fase propagasi mundur (*backpropagation*) dari *error* yang terkait. Selisih antara keluaran dan target merupakan kesalahan yang terjadi. Kesalahan tersebut dipropagasi mundur, dimulai dari garis yang berhubungan langsung dengan unit-unit dilayer keluaran;

iii) Fase modifikasi bobot.

Fase propagasi maju, Selama propagasi maju, sinyal masukan (X_1) dipropagasikan ke *layer* tersembunyi menggunakan fungsi aktivasi yang ditentukan. Keluaran dari unit tersembuyi (Z_j) tersebut selanjutnya dipropagasi maju lagi ke *layer* tersembunyi berikutnya dengan fungsi aktivasi yang telah ditentukan. Dan seterusnya hingga menghasilkan keluaran jaringan (Y_k).



Gambar 2.1 Algoritma Backpropagation

2.4 Bahasa pemrograman *Python*

Python berasal dari pengembangan bahasa pemrograman ABC oleh Guido van Rossum pada tahun 1990-an di Amsterdam. *Python* adalah bahasa pemrograman yang berorientasi objek dengan model skrip sehingga dapat dikatakan *Python* bersifat multifungsi karena dapat dipakai dalma pengembangan suatu perangkat lunak serta mampu berjalan pada semua sistem operasi.

Dalam penggunaannya *Python* memungkinkan para penggunanya untuk dapat lebih mengutamakan pengembangan aplikasi yang akan dibuat dikarenakan penulisan kode yang simpel dan mudah untuk diterapkan.

Karena bersifat *opensource* dan *freeware*, *Python* memiliki *library* yang dapat kembangkan dan dimengerti dengan mudah sehingga tidak ada batasan

dalam penggunaannya. Salah satu fitur yang tersedia pada *python* sebagai bahasa pemrograman dinamis yaitu *python* disempurnakan dengan adanya sistem manajemen memori otomatis.

2.5 Google Colaboratory

Sebuah aplikasi pengolahan data berbasis cloud dan gratis yang dikenal dengan nama Google Colaboratory atau Google Colab telah dikembangkan. Aplikasi ini dirancang menggunakan environment Jupyter dan mendukung berbagai library yang diperlukan dalam pengembangan teknologi kecerdasan buatan (AI), seperti yang telah dijelaskan oleh Wahyu pada tahun 2020.

Google Colab dibuat di atas environment Jupyter, sehingga mirip dengan Jupyter Notebook dan cara penggunaannya hampir sama. Satu-satunya perbedaan terletak pada media penyimpanan yang digunakan, yaitu Google Drive, dan alat ini berjalan pada sistem cloud. Selain itu, Google Colab juga menyediakan runtime Python 2 dan 3 yang telah dikonfigurasikan sebelumnya dengan berbagai *library*.

2.6 *Term Frequency – Inverse Document Frequency* (TF-IDF)

Term Frequency Inverse Text Frequency (TF-IDF) adalah sebuah metode yang digunakan untuk menghitung nilai frekuensi suatu kata dalam dokumen atau artikel dan sejauh mana kemunculan kata tersebut dalam beberapa publikasi. Algotima TF-IDF ini menentukan pentingnya sebuah kata dalam sebuah dokumen (Evan, 2014). Kamath pada tahun 2014 menjelaskan bahwa TF-IDF merupakan

salah satu algoritma yang sering digunakan dalam pengolahan data dalam skala besar.

TF menghitung seberapa sering sebuah kata muncul dalam dokumen. Sebuah kata mungkin bisa muncul berkali-kali dalam dokumen besar dibandingkan dengan yang kecil. Oleh karena itu, TF dihitung dengan membagi panjang dokumen. Dengan kata lain, TF dari sebuah kata dihitung dengan membaginya dengan jumlah kata dalam dokumen. TF dapat dihitung melalui persamaan berikut:

$$TF(t) = \frac{ft,d}{\sum t,d}$$

dimana *ft,d* merupakan frekuensi sebuah kata (*t*) muncul di dalam dokumen *d*, sedangkan $\sum t, d$ merupakan total keseluruhan kata yang terdapat di dalam dokumen d (Aizawa, 2002)

Untuk mengurangi dampak dari kata-kata umum seperti “apa,” “di,” dan seterusnya di korpus, makan digunkanlah TF-IDF. IDF memberikan bobot lebih pada kata-kata dengan frekuensi yang lebih tinggi atau lebih rendah. Sedangkan IDF diperoleh mealui persamaan berikut :

$$IDF_j = \log\left(\frac{|D|}{ft,D}\right)$$

|D| merupakan jumlah dokumen yang ada di dalam koleksi, sedangkan ft, merupakan jumlah dokumen dimana t muncul di dalam D (Salton & Buckley, 1988, Berger, et al, 2000). Dalam koleksi dokumen D, sebuah kata t dan dokumen individu d ϵ D

Kombinasi ini Metode TF dan IDF dikenal sebagai TF-IDF dan direpresentasikan secara matematis sebagai berikut:

$$TF\text{-}IDF(t,d,D) = TF(t,d) \times \log\left(\frac{D}{dft}\right)$$

di mana *t* menunjukkan istilah kata; *d* menunjukkan setiap dokumen; *D* mewakili kumpulan dokumen; dan d f_t menunjukkan jumlah dokumen dengan istilah *t* di dalamnya.

Jika *ft,d* bernilai besar dan *ft,D* bernilai kecil, maka nilai dari log (|D|/ft,D) cenderung tinggi dan juga TF-IDF(t) akan bernilai tinggi. Kata dengan nilai TF-IDF(t) yang tinggi berarti bahwa kata tersebut merupakan sebuah kata yang sangat penting di dalam dokumen *d* tetapi tidak di dalam kumpulan dokumen D.

2.7 *N-gram*

Dalam literaturnya William B. Cavnar and John M. Trenkle (2001) menyebutkan bahwa N-gram adalah sebuah potongan dari sebuah kalimat panjang dalam literaturnya mereka hanya menggunakan irisan yang berdekatan saja, satu potongan kalimat menghasilkan satu set N-gram yang tumpang tindih, di sistemnya mereka juga menyebutkan bahwa mereka menggunakan N-gram dengan ukuran yang berbeda terus-menerus, mereka juga menambahkan karakter kosong (dengan menggunakan karakter garis bawah "_" untuk mepresentasikan karakter kosong) di awal dan di akhir kalimat untuk membantu mencocokan kalimat di awal dan di akhir, contoh dari penelitian mereka menggunakan kata "TEKS" yang akan terdiri dari N-gram berikut:

uni-gram: T, E, K, S

bi-gram : _T, TE, EK, KS, S_

tri-gram : _TE, TEK, EKS, KS_, dan S_ _

Dapat disimpulkan bahwa untuk string berukuran *n* akan dimiliki pada n *unigram* dan n+1 *bigram*, n+1 *trigram* dan seterusnya. Penggunaan N-gram untuk *matching* kata memiliki keuntungan sehingga dapat diterapkan pada recovery pada input karakter ASCII yang terkena *noise*, interpretasi kode pos, *information retrieval* dan berbagai aplikasi dalam pemrosesan bahasa alami. Dengan model bahasa bigram memproses bahasa dengan memecah kalimat menjadi beberapa bagian dimana tiap bagian terdiri dari dua kata terurut dari kalimat. Jika suatu kalimat dinyatakan dalam notasi {w_1, w_2, w_3..., w_{i-1}, w_i} dan *i* menyatakan banyak kata dalam kalimat, maka nilai peluang bigram dapat dihitung melalui persamaan berikut,

$$\boldsymbol{P(w_i|w_{i-1}) = \frac{N(w_{i-1}, w_i)}{N(w_{i-1})}}$$

Notasi P(w_i|w_{i-1}) menyatakan nilai peluang bigram kata ke-i, didahului oleh kata ke (i-1). Notasi N(w_{i-1}, w_i) menyatakan banyaknya kemunculan pasangan kata ke-i didahului oleh kata ke (i-1) pada korpus. Notasi N($_{wi-1}$) menyatakan banyaknya kemunculan kata ke (i-1) pada korpus. Sebuah kalimat dapat dihitung nilai peluangnya menggunakan model statistik bigram dengan menerapkan aturan Chain yang dijabarkan dalam Persamaan berikut.

$$\boldsymbol{P(W_1^n) \approx \prod_{i=1}^{n} P(w_i/w_{i-1})}$$

Notasi (W_1^n) Menyatakan nilai peluang kalimat yang dihitung menggunakan model statistik bigram. Notasi n menyatakan banyaknya pasangan bigram kata dalam sebuah kalimat. Contoh penerapan persamaan adalah sebagai berikut.

W = Saya makan nasi padang

Maka peluang bigram dari kalimat tersebut dapat dihitung dengan rumus.

P(W) = P(makan | saya) * P(nasi | makan) * P(padang | nasi*)*

2.8 *Performance Evaluation Matrix* (PEM)

Ahmad Arif Budiman (2018) menggunakan *Performance Evaluation Measure* (PEM) atau dalam Bahasa Indonesia bisa disebut pengukuran evaluasi performa sebagai evaluasi pengukuran dalam penelitiannya, PEM sendiri adalah sebuah tahapan yang digunakan untuk mengukur performa suatu sistem. PEM dalam banyak kasus digunakan dalam *pelatihan data*, tujuannya untuk mengevaluasi model yang sudah dibuat. Ada banyak perhitungan untuk mendapatkan nilai PEM, biasanya diterapkan sebagai kombinasi atau juga secara parsial. Beberapa perhitungan dalam PEM antara lain :

a) Presisi.

   Presisi adalah tingkat ketepatan antara request pengguna dengan jawaban sistem;

   Rumus presisi (*pre*) : $\boldsymbol{pre} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}}$

b) Akurasi.

   Akurasi adalah perbandingan antara informasi yang dijawab oleh sistem dengan benar dengan keseluruhan informasi; dan

Rumus akurasi (*acc*) : $$\boldsymbol{acc} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+FP+TN+FN}}$$

c) *Recall*.

Recall adalah ukuran ketepatan antara informasi yang sama dengan informasi yang sudah pernah dipanggil sebelumnya.

Rumus *recall* (*rec*) : $$\boldsymbol{rec} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}}$$

PEM biasanya digambarkan dalam confusion matrix, yaitu berupa tabel yang berisi hasil pengujian model yang telah dibandingkan dengan *dataset*, terdiri dari kelas true dan false.

Tabel 2.2 Confusion Matrix

	Predict	***ed Class***
True Class	***Positive***	***Negative***
Positive	TP	FN
Negative	FP	TN

Keterangan :

TP (true positive) : contoh data bernilai positif yang diprediksi benar sebagai positif

TN (true negative) : contoh data bernilai negatif yang diprediksi benar sebagai negatif

FP (false positive) : contoh data bernilai negatif yang diprediksi salah sebagai positif

FN (false negative) : contoh data bernilai positif yang diprediksi salah sebagai negatif