Halo!

Sudah Siap Ikut Gemastik ? Semangat Ya ! :D

Pada cabang **Penambangan Data**, file makalah berisi inovasi yang berhubungan dengan penambangan data. Pada tahap ini, usahakan membuat yang terbaik ya! Karena makalah kamu akan digunakan sebagai senjata untuk menyingkirkan sainganmu sehingga kamu bisa berkesempatan menjadi finalis di Gemastik. Sebenarnya pada waktu itu tidak ada batasan terkait jumlah halaman, tetapi kami membatasinya hingga 30 halaman saja agar masih bisa dibaca dengan nyaman oleh para dewan juri.

Kalau kamu ingin bertanya atau diskusi, kamu bisa kontak kami di:

Douglas : kajix75 (LINE)

Pradipta : diptadipsi (LINE)

Makalah Penambangan Data Gemastik XIII/2020

DETEKSI EMOSI PADA TWEET BAHASA INDONESIA SECARA SEMI-SUPERVISED MENGGUNAKAN NEURAL GRAPH LEARNING UNTUK MENGEVALUASI PENYELENGGARAAN PENDIDIKAN SAAT PANDEMI

Disusun Oleh:

Tim Ramalin - 132000100251189

Douglas Raevan Faisal	(1706984562)
Rd Pradipta Gitaya Samiadji	(1706043361)

Dosen Pembimbing:

Rahmad Mahendra, S.Kom., M.Sc.

FAKULTAS ILMU KOMPUTER

UNIVERSITAS INDONESIA

DEPOK

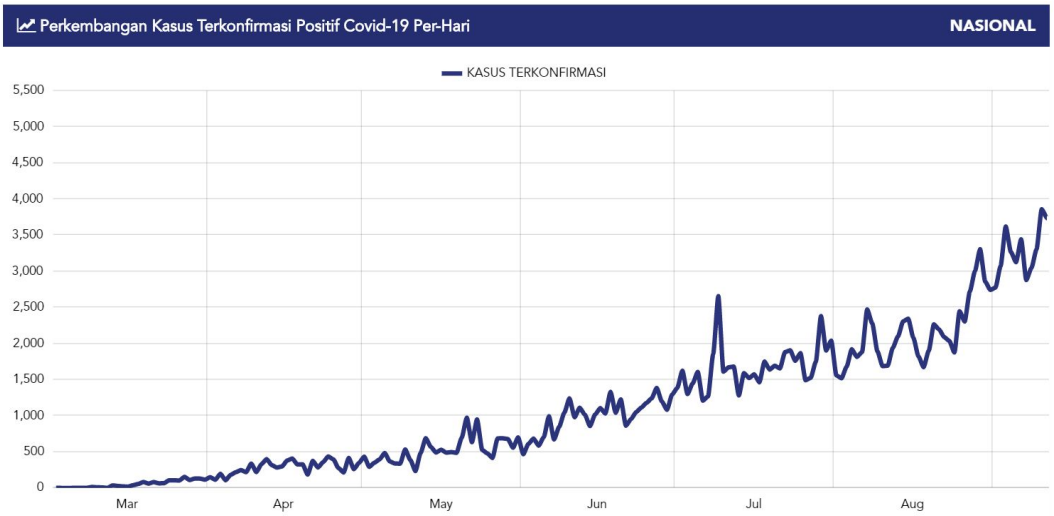
2020

DAFTAR ISI

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Merebaknya wabah *coronavirus disease* atau yang dinamakan COVID-19 secara global telah menimbulkan keresahan di tengah masyarakat dunia. Penyakit ini mulai diidentifikasi ketika wabah mulai terjadi di Wuhan, Tiongkok pada Desember 2019. Transmisi virus sampai ke Indonesia pada awal Maret 2020 dengan pengumuman dua kasus COVID-19 pertama di Indonesia. Hingga tanggal 11 September 2020, sudah terdapat 210.940 kasus positif COVID-19 di Indonesia dengan penambahan kasus terakhir sebanyak 3.737 kasus. Pemerintah telah melakukan berbagai upaya pada berbagai sektor untuk memperlambat laju penyebaran virus, salah satunya yaitu pada sektor pendidikan.



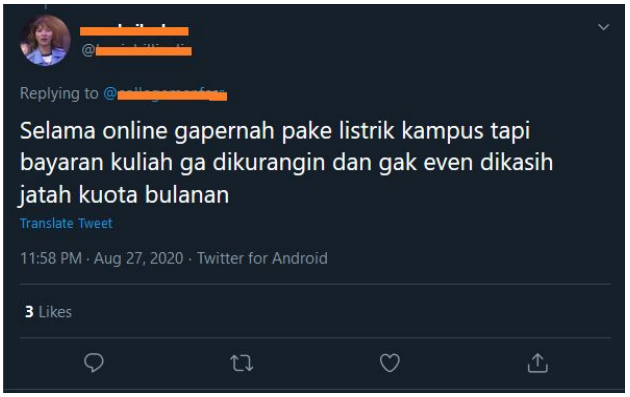
Gambar 1. Grafik Penambahan Kasus Konfirmasi Positif COVID-19 di Indonesia (covid19.go.id)

Sebagai upaya untuk menekan laju penyebaran COVID-19, Menteri Pendidikan dan Kebudayaan Nadiem Anwar Makarim mengeluarkan kebijakan melalui surat edaran nomor 4 tahun 2020 yang mewajibkan pelaksanaan kegiatan pembelajaran secara daring/jarak jauh untuk jenjang sekolah dan perguruan tinggi, dan kegiatan pembelajaran tatap muka ditiadakan. Kebijakan ini disebut dengan Pendidikan Jarak Jauh (PJJ).

Salah satu poin penting dijelaskan pada poin 2.a surat edaran nomor 4 tahun 2020 yang mengatakan "*Proses belajar dilakukan dari rumah melalui pembelajaran daring/jarak jauh untuk memberikan pengalaman belajar yang bermakna bagi siswa, tanpa terbebani tuntutan menuntaskan seluruh capaian kurikulum untuk kenaikan kelas maupun kelulusan*". Tidak hanya itu, masyarakat juga diharapkan memaksimalkan fasilitas teknologi untuk aktivitas belajar. Kebijakan ini ditempuh sebagai langkah darurat dengan tujuan agar pelaksanaan pembelajaran dapat tetap berjalan dengan baik di masa pandemi.

Kebijakan pelaksanaan PJJ ini menimbulkan pro dan kontra di masyarakat, karena penetapannya bersifat cukup mendadak dan tidak semua ranah ilmu siap untuk menjalankan pembelajaran secara daring, terutama untuk rumpun ilmu kesehatan dan rumpun ilmu sains yang masih memerlukan kegiatan tatap muka untuk aktivitas seperti praktikum dan riset yang membutuhkan perangkat khusus. Berbagai reaksi masyarakat terutama dari kalangan pelajar dan pengajar terkait adanya kebijakan ini pun dapat ditemukan melalui berbagai kanal media sosial seperti Facebook dan Twitter. Reaksi yang diberikan pun beragam, dari reaksi menyambut kebijakan ini dengan gembira, hingga reaksi lainnya seperti marah, sedih, cinta hingga takut akan dampak

terhadap pelaksanaan pendidikan di masa depan. Ada pula masyarakat yang enggan berkomentar terkait kebijakan pelaksanaan PJJ. Salah satu pihak yang terdampak PJJ adalah mahasiswa. Pada Gambar 2 ditunjukkan sebuah reaksi mahasiswa yang muncul di mana mahasiswa mengeluhkan tidak sedang menggunakan fasilitas kampus namun biaya perkuliahan tidak dikurangi. Secara intuisi manusia, dapat ditangkap reaksi tersebut adalah reaksi amarah.



Gambar 2. Reaksi amarah salah seorang mahasiswa terkait pelaksanaan PJJ

Berbagai respon masyarakat di kanal media sosial dapat dideteksi emosinya melalui pendekatan *predictive analytics* dengan memanfaatkan teknologi kecerdasan buatan. Pendekatan ini digunakan karena besarnya ukuran data sehingga sulit untuk diolah dan dianotasi satu per satu oleh manusia. Menggunakan pendekatan *predictive analytics* pada konten teks di media sosial memunculkan potensi baru dari teknologi kecerdasan buatan untuk membantu memberikan estimasi dan gambaran yang cukup akurat terhadap respon masyarakat dengan mengklasifikasikan emosi yang terindikasi, khususnya pada konteks PJJ di Indonesia.

Kebijakan PJJ memang tergolong baru dan masih perlu dilakukan evaluasi untuk mengetahui seberapa baik kinerjanya antara sebelum dengan sesudah diterapkan kepada masyarakat, serta perbaikan yang dapat dilakukan untuk penyelenggaraan kegiatan PJJ yang lebih baik di masa depan. Untuk mengetahui seberapa baik tingkat penerimaan masyarakat terkait pelaksanaan PJJ dapat dilakukan *emotion analysis* terhadap respon masyarakat di kanal media sosial. Hasil deteksi emosi menggunakan pendekatan *predictive analytics* dapat dimanfaatkan oleh sebagai bahan umpan balik serta evaluasi untuk perbaikan pelaksanaan PJJ di Indonesia. Tidak hanya untuk PJJ, namun perbaikan pelaksanaan pendidikan secara menyeluruh di Indonesia sebagai upaya untuk mencapai visi menuju Indonesia Maju.

Melihat potensi serta permasalahan ini, kami dari tim Ramalin mengusulkan sebuah solusi berupa sistem untuk mendeteksi jenis emosi masyarakat terutama pelajar di Indonesia. Sistem ini mengambil teks respon masyarakat yang ingin diprediksi. Kemudian melalui pengolahan bahasa manusia dan *deep learning*, sistem akan memprediksi bagaimana jenis emosi dari teks yang didapatkan. Sistem akan memprediksi apakah teks tersebut mengandung emosi senang, cinta, takut, sedih, atau marah. Deteksi dilakukan menggunakan komputer, sehingga prosesnya cepat dan pihak berwenang dapat mengambil tindakan secara cepat dan tanggap.

1.2 Rumusan Masalah

Urgensi utama yang hendak dibahas pada dokumen penelitian ini adalah bagaimana model *predictive analytics* yang dapat digunakan untuk menghasilkan prediksi emosi berdasarkan *tweet* secara akurat. Selain itu, pada penelitian ini tim peneliti juga tertarik untuk membahas perbandingan performa prediksi menggunakan *supervised learning* dan *semi-supervised learning* dengan memanfaatkan *tweet* Bahasa Indonesia.

1.3 Tujuan, Manfaat, dan Potensi Keberlanjutan

Berdasarkan latar belakang di atas, penelitian ini hadir dengan tujuan sebagai berikut:

- Mendukung visi menuju Indonesia Maju pada aspek pendidikan sekaligus membantu mewujudkan keberhasilan program pendidikan dari Kemendikbud untuk perbaikan mutu pendidikan di Indonesia terutama pelaksanaan pendidikan saat masa pandemi memanfaatkan kecerdasan buatan.
- Memberikan solusi bagi pemerintah dan pihak berwenang baik tingkat pusat maupun daerah untuk identifikasi kebutuhan masyarakat terutama pelajar terkait pelaksanaan pendidikan saat pandemi melalui pendekatan *predictive analytics*.
- Memaparkan model - model yang dapat digunakan untuk melakukan prediksi jenis emosi manusia dengan menggunakan *tweet* pengguna media sosial Twitter.
- Memperkenalkan kombinasi pendekatan *semi-supervised learning*, *deep learning*, dan *Natural Language Processing* (NLP) dalam merancang model prediksi menggunakan teks terutama Bahasa Indonesia dimana saat ini masih relatif sedikit orang yang meneliti dikarenakan struktur Bahasa Indonesia yang cukup kompleks.
- Mendukung salah satu tujuan pada *Sustainable Development Goals* (SDGs) yang dibuat oleh Majelis Umum PBB nomor 4, yaitu "*ensure inclusive and equitable quality education and promote lifelong learning opportunities for all*"

Dengan hadirnya penelitian ini, diharapkan dapat memberikan manfaat dengan menghasilkan model prediksi emosi yang akurat sehingga dapat dimanfaatkan untuk mempercepat pengenalan kebutuhan masyarakat terutama pelajar terhadap pendidikan di masa pandemi dan masa depan. Selain itu, diharapkan penelitian ini dapat mempermudah upaya perbaikan pelaksanaan pendidikan agar lebih efektif, efisien, dan merata untuk membentuk insan muda yang prestatif dan berkarakter demi mewujudkan visi menuju Indonesia Maju.

1.4 Relevansi dan Pentingnya Penelitian

Penelitian ini membawa semangat yang sama dengan tema Gemastik XIII yaitu "Teknologi Informasi dan Komunikasi untuk Indonesia Maju" dari aspek pendidikan. Sejak pandemi COVID-19 melanda di Indonesia, berbagai kebijakan darurat terkait pendidikan telah diterapkan oleh Mendikbud Nadiem Anwar Makarim agar pelaksanaan pendidikan dapat tetap berjalan dengan baik. Akan tetapi untuk mewujudkan hal ini secara sempurna, perlu melalui banyak kajian dan umpan balik masyarakat, terutama dari aktor pelaksana pendidikan saat ini. Salah satu hal yang berpotensi dijadikan sebagai bahan evaluasi adalah informasi terkait reaksi masyarakat terhadap pelaksanaan pendidikan saat ini melalui pendekatan *predictive analytics*.

Penelitian ini sangatlah penting, karena dengan *predictive analytics* terhadap emosi, pemerintah dan pihak berwenang dapat mengetahui reaksi atau emosi pelajar terhadap pelaksanaan pendidikan di masa pandemi COVID-19 saat ini, sehingga dapat dipergunakan sebagai bahan evaluasi untuk perbaikan kebijakan pendidikan di masa depan.

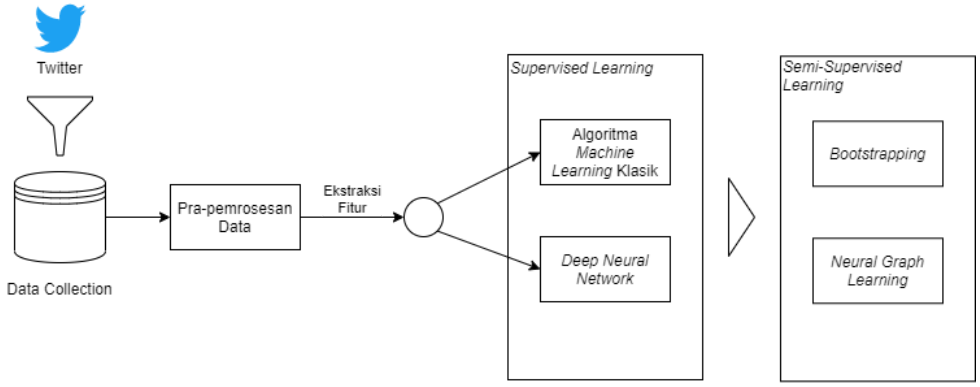
1.5 Batasan Penelitian

Berikut ini adalah batasan ruang lingkup serta implementasi pada penelitian ini:

- Data yang digunakan adalah *tweet* dari media sosial Twitter yang memiliki unsur dominan bahasa Indonesia. *Tweet* tidak perlu 100% berbahasa Indonesia (boleh mengandung campuran dengan bahasa lain). Misalnya sebuah *tweet* memiliki proporsi bahasa Jawa sebanyak 30% dan bahasa Indonesia 70%.

- Data yang digunakan adalah *tweet* yang secara umum bersifat tidak baku (tidak mengikuti kaidah EYD) sehingga untuk mendapatkan hasil yang lebih akurat perlu adanya pengumpulan data dan jumlah *annotator* yang lebih banyak di masa depan.
- Proses anotasi dibatasi satu buah *tweet* hanya dapat dikategorikan sebagai satu kelas emosi, atau tidak dikategorikan sama sekali. Hal ini membuat tidak ada kasus dimana sebuah *tweet* digolongkan pada lebih dari satu jenis emosi.
- Metode yang dipaparkan hanya dapat melakukan prediksi terhadap lima jenis emosi, yaitu *happy* (senang), *love* (cinta), *sadness* (kesedihan), *anger* (amarah), dan *fear* (takut). Penentuan lima kelas emosi ini terinspirasi dari penelitian Saputri et al. (2018).

1.6 Alur Penelitian



Gambar 3. Diagram Alur Penelitian

Alur penelitian ini dapat dibagi menjadi tiga tahapan besar, yaitu (1) pengumpulan data dan pra-pemrosesan data, (2) eksperimen pendekatan *supervised learning* menggunakan algoritma *machine learning* klasik dan algoritma *deep neural network*, serta (3) eksperimen pendekatan *semi-supervised.*

Pada tahap pengumpulan data, tim peneliti menggunakan dua jenis sumber data, yaitu data primer dan data sekunder. Data primer merupakan data *tweet* yang langsung dikumpulkan dan dianotasi oleh tim peneliti. Hal ini dilakukan karena isu terkait pendidikan di masa pandemi saat ini yang masih cukup baru, dan tim peneliti belum menemukan dataset *tweet* yang telah dianotasi. Data sekunder merupakan data *tweet* yang telah dikumpulkan dan dianotasi dari penelitian rujukan. Tujuan penggunaan data sekunder ini untuk memanipulasi dataset *training* agar algoritma dapat berfokus pada emosi namun tetap berada pada cakupan konteks penelitian, yaitu pendidikan di masa pandemi. Berikutnya, data yang telah terkumpul dan dianotasi, akan melalui tahapan pra-pemrosesan untuk mengekstrak fitur untuk dijadikan sebagai masukan pada model prediksi.

Selanjutnya, beberapa algoritma akan diujikan pada dataset hasil pra-pemrosesan. Secara garis besar, terdapat dua jenis algoritma yang akan diuji, yaitu algoritma *machine learning* klasik (contoh: Naive Bayes, Support Vector Machine, Decision Tree) dan algoritma *deep neural network*. Tidak hanya menggunakan algoritma *machine learning klasik*, tim peneliti juga ingin memanfaatkan algoritma *deep neural network* untuk membandingkan performa yang didapatkan.

Pada tahapan akhir, tim peneliti akan menggunakan model yang terbaik pada pendekatan *supervised learning* untuk diuji pada pendekatan *semi-supervised learning*. Dua pendekatan yang akan diuji yaitu *Bootstrapping* dan *Neural Graph Learning*. Tim peneliti memilih *bootstrapping* karena banyaknya implementasi dari pendekatan ini pada konteks *semi supervised learning*, dan juga memilih *Neural Graph Learning* karena unsur kebaruan (diperkenalkan pada tahun 2018) dan potensi dari pendekatan ini yang diklaim dapat meningkatkan performa model baik dalam kondisi *supervised learning* maupun *semi supervised learning*.

BAB 2
PENGUMPULAN DAN ANALISIS DATA

2.1 Kebutuhan Data

Agar penelitian ini dapat berjalan dengan baik, tim peneliti membutuhkan data teks yang relatif pendek, berbahasa Indonesia walaupun peneliti tidak mewajibkan teks harus sesuai dengan ketentuan EYD (Ejaan Yang Disempurnakan), memiliki kandungan emosi, serta memiliki kalimat yang mengandung konteks ranah pendidikan. Untuk menyelesaikan permasalahan ini, tim peneliti pada akhirnya memiliki menggunakan teks *tweet* pada Twitter sebagai sumber data primer.

Tim peneliti memilih *tweet* sebagai sumber data primer dikarenakan teks pada sebuah *tweet* memiliki panjang maksimum 280 karakter. Meskipun memiliki maksimal 280 karakter, umumnya pengguna Twitter cenderung menuliskan teks relatif pendek dengan rentang hanya 150-200 karakter. Teks yang pendek akan mempercepat *training* dan proses data pada model pembelajaran mesin yang dirancang. Selain itu, berbeda dengan platform media sosial lainnya, hampir setiap *tweet* pengguna di Twitter sepenuhnya bersifat publik dan dapat diambil. Ini merupakan nilai tambah yang sangat besar untuk keperluan analisis data.

Saat ini Twitter telah menyediakan API (*Application Programming Interface*) resmi yang memungkinkan para peneliti melakukan kueri kompleks untuk mengambil data *tweet* dan informasi relevan lainnya berdasarkan kriteria tertentu. Tidak hanya API yang bersifat resmi saja, saat ini cukup banyak API dan perangkat lunak pihak ketiga yang menunjang keperluan analisis data menggunakan *tweet*.

2.2 Perangkat Pengumpulan Data

Khusus untuk pengumpulan data primer yang terkait dengan bidang pendidikan di masa pandemi, tim peneliti memanfaatkan akses secara gratis ke Twitter API. Untuk *runtime* Python, tersedia *library* Tweepy yang merupakan sebuah Twitter API Wrapper. Dengan menggunakan *library* tersebut, tim peneliti dapat mengotomatisasi proses pengumpulan *tweet* dan membentuk *pipeline* pemrosesan data berikutnya. Peneliti mengadaptasikan skrip Python yang dipublikasi pada artikel Medium oleh Griffin Leow (2019) dengan menyesuaikan kata kunci pencarian.

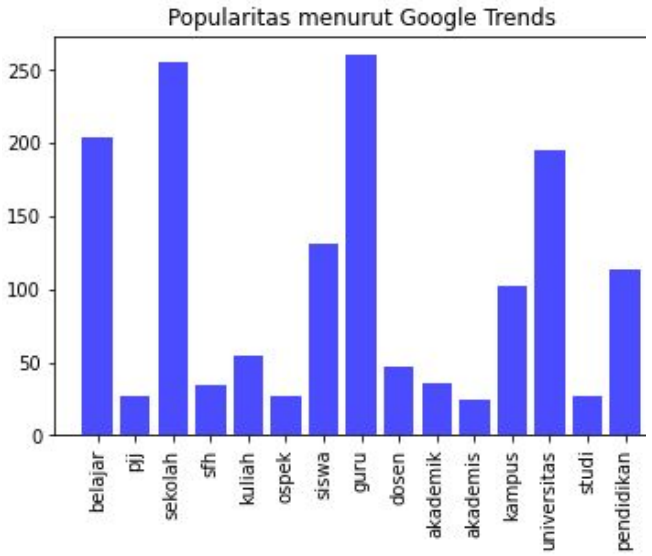
2.3 Proses Pengumpulan Data

Tim peneliti mengumpulkan data secara mandiri. Secara umum, terdapat dua metode untuk pengumpulan data *tweet*, yaitu metode *search* dan metode *stream.* Metode *search* memanfaatkan Twitter Search API untuk memperoleh *tweet-tweet* yang terdahulu hingga 7 hari ke belakang, sementara itu metode *stream* memanfaatkan Twitter Stream API dengan memperoleh sampel *tweet* secara *live* dengan *rate sampling* maksimum 1% dari keseluruhan *tweet* di dunia. Tim peneliti memutuskan untuk menggunakan metode *search* dengan pertimbangan efisiensi waktu pengumpulan data. Dengan metode tersebut, diestimasi peneliti dapat mengumpulkan hingga 180.000 *tweet* per jam (terdapat *limit* API *request* dalam 15 menit sebanyak 450 kali *request*, dan jumlah *tweet* yang diperoleh hingga 100 *tweet* pada setiap *request*).

Berikutnya, tim peneliti menentukan kata-kata dalam Bahasa Indonesia yang dapat dikaitkan dengan bidang pendidikan dan konteksnya di masa pandemi. Tim peneliti berhasil mengumpulkan 17 kata kunci yang dapat mengindikasikan bahwa sebuah *tweet* sedang membahas tentang pendidikan. Berikutnya adalah menentukan cakupan tanggal pengumpulan data. Tim peneliti melakukan pengumpulan data pada tanggal 30 Agustus 2020, dan karena adanya limitasi Twitter API untuk pencarian *tweet* hingga 7 hari ke belakang, maka tim peneliti mendefinisikan tanggal pengumpulan data dimulai dari 22 Agustus 2020. Selain itu, untuk konteks deteksi emosi, peneliti memutuskan untuk tidak mengandung *retweet*, karena *tweet* tersebut bukan merupakan ekspresi langsung dari diri pribadi dan bisa menyebabkan *redundancy* pada data yang dapat mempengaruhi

proses *training*. Namun, penambahan eksklusi *retweet* pada *query* seperti di bawah tidak mengeksklusi *retweet* yang bersifat *in-text* seperti *"RT @username: ga kebayang ospek taun ini..."*. Eksklusi *in-text retweet* dilakukan sebelum data disimpan ke sebuah file CSV dengan menggunakan *regular expression* (Regex) *pattern matching*. Peneliti juga membatasi bahasa yang digunakan hanya Bahasa Indonesia.

Dalam menentukan kata kunci yang akan digunakan, tim peneliti memiliki beberapa hal yang menjadi pertimbangan. Salah satu pertimbangan utama adalah seberapa besar tingkat popularitas serta relevansi kata kunci yang akan digunakan terhadap konteks penelitian. Sebagai langkah awal, tim peneliti menggunakan Google Trends untuk mencari tahu seberapa populer kata kunci dalam lima hal, yakni pencarian web, pencarian gambar, pencarian berita, pencarian belanja, serta pencarian video oleh para pengguna selama 1 bulan terakhir (30 Juli - 28 Agustus 2020). Berikut adalah data rerata popularitas beberapa kata kunci menurut Google Trends.



Gambar 4. Perbandingan popularitas kata kunci menurut Google Trends

Berdasarkan data di atas, terlihat lima kata kunci paling populer adalah belajar, sekolah, guru, universitas, dan siswa. Tim peneliti sangat yakin mendapatkan cukup banyak data dari kelima kata kunci utama tersebut. Selain menggunakan Google Trends, tim peneliti juga melihat apa yang menjadi *trending* dari media sosial Twitter, serta bertanya kepada orang disekitar tempat tinggal tim peneliti seputar topik apa sajakah yang menurut mereka populer di sekolah, di universitas tempat menjalankan studi hingga lingkungan pertemanan mereka. Tim peneliti berhasil mendapatkan kata kunci baru yang berpotensi memperkuat penelitian. Berikut adalah *query* final yang digunakan oleh tim peneliti untuk melakukan pengambilan data.

```
"(belajar OR pjj OR sfh OR sekolah OR kuliah OR ospek OR siswa OR guru OR dosen OR
akademik OR akademis OR kampus OR universitas OR studi OR pendidikan OR
perkuliahan OR pelajaran) since:2020-08-22 until:2020-08-30 -is:retweet lang:id"
```

2.4 Pra Pemrosesan Data

Langkah pertama pada pra-pemrosesan data yaitu melakukan *random sampling* pada dataset *tweet* yang berhasil terkumpul. Langkah ini dilakukan hanya untuk data primer, yaitu data *tweet* pendidikan agar proses anotasi tidak memakan waktu. Target yang diharapkan oleh tim peneliti yaitu memperoleh 10.000 *tweet* untuk berikutnya dianotasi.

Langkah berikutnya adalah *replacement* informasi pada *tweet* yang bersifat sensitif. Untuk kebutuhan ini, terdapat dua jenis informasi yang berhasil di-*replace* dengan sebuah tanda khusus, yaitu *username* Twitter dan URL atau link yang ada pada sebuah *tweet*. Tim peneliti sempat bereksperimen untuk melakukan *replacement* pada nomor telepon, tetapi terdapat kendala di mana Regex *pattern matching* juga menghapus angka 19 pada teks seperti "COVID-19", sementara angka tersebut muncul dalam frekuensi yang cukup tinggi dan peneliti merasa angka ini dibutuhkan untuk memperjelas konteks yang dibahas pada *tweet* tersebut dan mendampingi konteks utama, yaitu pendidikan.

Setelah dilakukan *replacement* pada *username* dan URL, berikutnya tim peneliti melakukan anotasi emosi secara manual oleh 2 orang. Tim peneliti membagi tugas, kemudian menganotasi dengan mengikuti panduan anotasi emosi oleh Shaver et al. (2001). Terdapat lima jenis label emosi yang digunakan, yaitu *happy* (senang), *love* (cinta), *sadness* (kesedihan), *anger* (amarah), dan *fear* (takut). Berikut adalah beberapa contoh *tweet* yang diasosiasikan dengan masing-masing label.

Tabel 1. Contoh *tweet* mentah untuk masing-masing label

Label	Contoh Tweet
Happy	" <USERNAME> semoga semua kembali normal dan beraktivitas seperti biasa. semoga rejekiku lancar dan sehat selalu ✨ semoga IPK ku naik" "in kelas di zoom gajadi belajar gais 🤣 jd kelas stand up comedy" "bahagia bgt walaupun cuman ngeliat temen kampus lewat gmeet, sehat sehat ya kalian."
Love	" <USERNAME> hv a nice dayy!!! jan lupa sarapan, semangatt pjj nyaa!!□□" " <USERNAME> Pagi juga, semangat PJJ untuk kamu yang disana" "Habis ini mau niat nyari pacar deh, mumpung kuliah online,"
Anger	"pengin istirahat dari layar gadget buat satu hari aja, aku capek banget, mau gini besok tapi aku minggu ada pjj:(" "<USERNAME> Selama online gapernah pake listrik kampus" "ya allah guru ada ada aj si😭 klo zoom tiap minggu untuk apa diadakannya minggu ganjil genap😭"
Sadness	" <USERNAME> gk bsa tdur lgi kan pjj sep,,, 💔" " <USERNAME> —c Ikutan banyak GA karena sebutuh itu kaaa, aku ga les offline maupun online, penghasilan ortu juga terdampak akibat pandemi ini.... ditambah skrg pjj yg membutuhkan kuota banyak jadi tabunganku aku utamain dulu buat beli kuota drpd buat beli buku :((. —c" "dibanding sekolah online, yang gapunya sarana untuk ikutan juga kasian, mereka minimal punya hp untuk zoom meeting, etc."

Tabel 2. Contoh *tweet* mentah untuk masing-masing label (lanjutan)

Fear	"kepikiran tar pas udah mulai kuliah, terus ditanya, 'Selama pandemic dan libur ini kalian udah ngelakuin apa aja' gue harus jawab apa....."
	" <USERNAME> Sebenernya pjj ini jadi tantangan banget buat aku karena aku kadang gak begitu ngerti sama materinya tapi dituntut untuk paham. Susah sekali rasanya untuk pahamin sendiri. Tapi aku suka dengan prosesnya, lihat cara kerjanya dan penyelesaiannya. Aku akan sangat terbantu dengan ini"
	"gerakan 2020 menolak sekolah offline tapi butuh pisan euy"

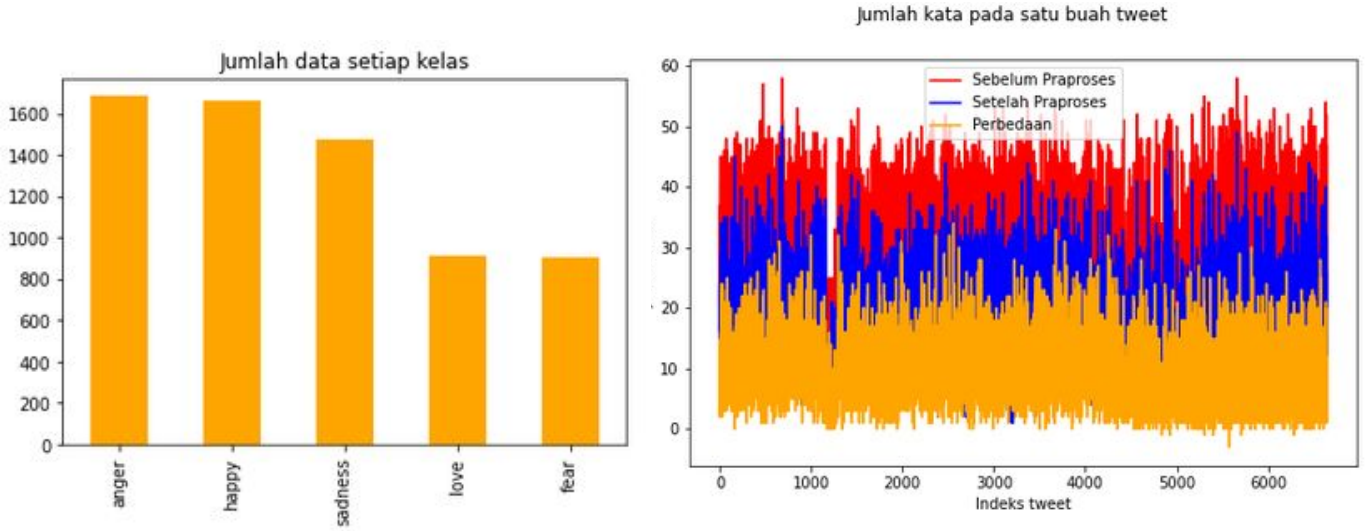
Langkah terakhir yang diterapkan dari tahap ini yaitu pemrosesan teks. Beberapa aktivitas *Natural Language Processing* (NLP) diimplementasikan pada tahap ini, yaitu *lowercasing*, pembuangan *stopwords*, dan tokenisasi. Terdapat satu aktivitas yang tidak mampu dilakukan tim peneliti, yaitu *stemming*. Eksperimen untuk *stemming* sebenarnya dapat dilakukan dengan bantuan *library* Sastrawi karena mendukung untuk *stemming* Bahasa Indonesia. Namun, dikarenakan proses tersebut memakan waktu yang lama (rerata 15 detik/*tweet*), sedangkan data yang dimiliki tim peneliti cukup banyak, maka peneliti memutuskan untuk tidak menggunakan *stemming* untuk penelitian ini.

2.5 Analisis Data Terkumpul

Setelah data berhasil dikumpulkan, tim peneliti berhasil mendapatkan data sebanyak 255.071 *tweet*. Jumlah *tweet* tersebut diperoleh dalam kurun waktu 3 jam dan telah disaring sehingga tidak ada *tweet* yang merupakan *retweet* (*tweet* tersebut adalah merupakan teks yang dibuat pengguna Twitter secara personal). Langkah selanjutnya yaitu melakukan *sampling* dari data yang sudah terkumpul, dan diperoleh 10.000 *tweet* untuk berikutnya dianotasi oleh tim peneliti secara manual. Untuk proses ini, tim peneliti hanya berfokus pada teks *tweet* sebagai fitur untuk deteksi emosi, maka dari itu atribut *tweet* lainnya seperti *username* dan *timestamp* dihapus dari dataset. Atribut teks berikutnya diolah untuk mengganti *username* dan URL menjadi label "<USERNAME>" dan "<URL>".

Dataset yang telah dianotasi mengikuti format dari Saputri et al. (2018), di mana dataset tersebut hanya terdiri dari 2 kolom atribut, yaitu *tweet* serta label emosi. Agar proses dapat berjalan dengan efisien, tim peneliti memberikan batasan satu buah *tweet* hanya dapat digolongkan menjadi satu kelas emosi atau tidak sama sekali, sehingga jika terdapat *tweet* yang mengandung lebih dari satu emosi (menurut intuisi manusia), maka *tweet* tersebut akan dikategorikan ke kelas emosi yang dominan. Selain itu, tim peneliti menggunakan aturan bahwa *tweet* yang dikategorikan adalah teks yang dibuat secara personal dan mengandung emosi dari sudut pandang orang yang menulis, sehingga *tweet* seperti iklan, berita, dan infografis tidak termasuk dalam lima kelas emosi. Tim peneliti berhasil mendapatkan sebanyak 6644 teks data akhir yang dikategorikan ke dalam lima buah kelas, dan distribusi yang didapatkan mengikuti distribusi normal, sehingga cocok untuk dianalisis lebih lanjut. Kelas yang mendominasi adalah *anger* dan *happy*.

Setelah melakukan pra-pemrosesan data, tim peneliti tertarik melihat perbandingan jumlah kata pada sebuah *tweet* sebelum dan sesudah pra-pemrosesan. Setelah melakukan pra-proses, setiap teks *tweet* menjadi lebih pendek. Ini menjadi nilai tambah yang besar, karena sangat berpotensi menurunkan besarnya biaya komputasi serta berpotensi meningkatkan akurasi dari model yang dirancang pada penelitian ini. Berikut ini adalah visualisasi dari distribusi data yang ada.



Gambar 5. (a) Perbandingan teks berbagai kelas (*annotated)* (b) Perbandingan jumlah kata pada sebuah *tweet* sebelum dan sesudah dilakukan pra-pemrosesan data

Tabel 3. Contoh perbandingan *tweet* sebelum pra-pemrosesan dan setelah pra-pemrosesan

Tweet Sebelum Pra Proses	Tweet Setelah Pra Proses
“in kelas di zoom gajadi belajar gais jd kelas stand up comedy”	“in kelas zoom gajadi belajar gais jd kelas stand up comedy”
“bahagia bgt walaupun cuman ngeliat temen kampus lewat gmeet, sehat sehat ya kalian.”	“bahagia bgt cuman ngeliat temen kampus gmeet sehat sehat ya”
“<USERNAME> Selama online gapernah pake listrik kampus”	“online gapernah pake listrik kampus”
“ <USERNAME> Sebenernya pjj ini jadi tantangan banget buat aku karena aku kadang gak begitu ngerti sama materinya tapi dituntut untuk paham. Susah sekali rasanya untuk pahamin sendiri. Tapi aku suka dengan prosesnya, lihat cara kerjanya dan penyelesaiannya. Aku akan sangat terbantu dengan ini”	“sebenernya pjj tantangan banget kadang gak ngerti materinya dituntut paham susah pahamin suka prosesnya lihat kerjanya penyelesaiannya terbantu“
“dibanding sekolah online, yang gapunya sarana untuk ikutan juga kasian, mereka minimal punya hp untuk zoom meeting, etc.”	“dibanding sekolah online gapunya sarana ikutan kasian minimal hp zoom meeting etc”

BAB 3
METODE PENAMBANGAN DATA

3.1 Ekstraksi Fitur Teks

3.1.1 Ekstraksi Fitur Bag-of-Words (BoW)

Fitur BoW adalah fitur yang berisi informasi frekuensi kemunculan suatu kata dalam sebuah dokumen. Banyaknya kata unik yang terbentuk tergantung dari data yang digunakan. Pada penelitian ini, tim peneliti akan mengambil sebanyak 2.000 kata unik yang memiliki frekuensi kemunculan tertinggi dari keseluruhan data *tweet* yang digunakan, sehingga *output* yang dihasilkan adalah 2000 kata unik beserta frekuensi kemunculannya dari setiap *tweet* dalam bentuk matriks menggunakan fungsi countVectorizer dari *library* scikit-learn.

3.1.2 Ekstraksi Fitur Sentimen Leksikon (Kamus Sentimen)

Sebuah teks Bahasa Indonesia dapat mengandung sentimen positif maupun negatif, tergantung dari susunan kata serta leksikon yang yang membentuknya. Pada penelitian ini, data leksikon yang digunakan berasal dari penelitian Vania et. al (2014). Leksikon terdiri dari 415 kata positif dan 581 kata negatif. Untuk setiap *tweet* akan dihitung jumlah kata positif dan negatifnya dalam bentuk *array* 1 dimensi. Berikut adalah ilustrasi bagaimana ekstraksi fitur sentimen leksikon bekerja.

Tabel 4. Ilustrasi Fitur Leksikon

Contoh teks	Positif / Negatif	Hasil
Hari ini aku merasa **bahagia**	1 / 0	[1,0]
Aku **cinta** kamu, kamu **benci** aku	1 / 1	[1,1]
Dasar tukang **bohong**!	0 / 1	[0,1]

3.1.3 Ekstraksi Fitur Part-of-Speech Tag (POS Tag)

Part-of-speech Tagging (POS Tag) merupakan proses pemberian penanda kelas sintaktik pada setiap kata di dalam sebuah korpus dengan kode tertentu. POS Tag dapat digunakan untuk mengenali emosi pada *tweet*. Pada penelitian ini, akan dihitung kemunculan kata sifat (Kode: JJ) dan kata negasi (Kode: NEG) setiap *tweet* menggunakan *pre-trained* POS Tag Bahasa Indonesia dari penelitian Dinakaramani et al (2014).

3.1.4 Ekstraksi Fitur Ortografi

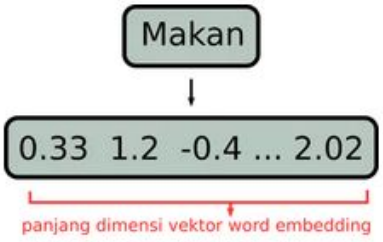
Ortografi adalah seperangkat aturan yang menjelaskan penggunaan ejaan secara tertulis. Fitur ortografi menjadi penting karena penggunaan tanda baca, huruf kapital dan panjang karakter menandakan adanya emosi. Pada percobaan ini, fitur ortografi yang digunakan untuk mengenali emosi pada *tweet* adalah jumlah huruf kapital, jumlah tanda seru, jumlah huruf dan panjang karakter (termasuk tanda baca dan karakter spasi) pada sebuah *tweet*. Berikut adalah ilustrasinya.

Tabel 5. Ilustrasi Fitur Ortografi

Contoh teks	Kapital	Tanda Seru	Jumlah Kata	Panjang Teks	Hasil
PJJ terus nih kapan offline lagi	3	0	6	32	[3,0,6,32]
Pandemi bikin MISKIN aja!!!	7	3	4	27	[7,3,4,27]

3.2 *Word Embedding*

Word embedding adalah teknik *deep learning* yang memetakan makna semantik dari sebuah teks ke dalam ruang geometris berdimensi tinggi (biasanya vektor berdimensi 100 atau lebih). Pada dasarnya, dengan metode ini, peneliti dapat mengubah kata menjadi sebuah vektor yang berisi angka - angka yang cukup kecil namun dapat mengandung informasi yang banyak dengan cara mengubah sebuah string menjadi *one-hot encoding*. Selain itu dengan *word embedding*, vektor dapat mengenali konteks makna serta seberapa dekat makna sebuah kata dengan kata lain. Contohnya seperti "marah" dan "kesal" memiliki makna yang berdekatan ketimbang kata "marah" dengan "bahagia" yang maknanya bertolak belakang. Teknik ini sudah dikenal mampu untuk dipergunakan sebagai model klasifikasi teks.



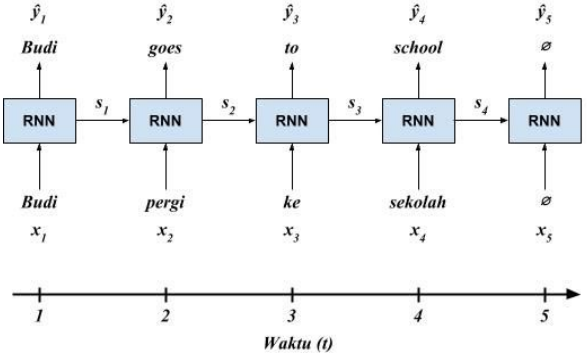
Gambar 6. Ilustrasi kata bahasa Indonesia dalam representasi vektor *word embedding*

Penggunaan *word embedding* pada beberapa tahun belakangan ini lebih sering digunakan untuk klasifikasi teks berbahasa Inggris seperti halnya penelitian yang dilakukan oleh Heriz et al (2017) dan Vora et al. (2017). Meskipun begitu, penggunaan *word embedding* masih jarang digunakan untuk Bahasa Indonesia, sehingga tim peneliti tertarik untuk melakukan eksperimen menggunakan *word embedding* sebagai fitur dalam perancangan model klasifikasi emosi.

Pada penelitian ini, tim peneliti menggunakan layer *embedding* dari TensorFlow untuk pembuatan fitur *word embedding* dari data. Tim peneliti bereksperimen menggunakan *word embedding* berdimensi 50, 100, 200, dan 300 yang kemudian dievaluasi serta dipilih nilai dimensi yang menghasilkan performa terbaik.

3.3 *Recurrent Neural Network & Convolutional Neural Network*

RNN (*Recurrent Neural Network*) adalah jenis *deep neural network* yang melakukan pemrosesan data secara berulang-ulang. Biasanya pada RNN data diproses secara sekuensial. RNN masuk dalam kategori *deep learning* karena data diproses melalui banyak lapisan (*layer*). RNN telah banyak digunakan dalam pengolahan teks, dimulai dari klasifikasi teks hingga pengembangan mesin penerjemah bahasa. RNN digunakan karena kemampuannya untuk "mengingat" data yang masuk pada untuk setiap tahap pemrosesan.



Gambar 7. Ilustrasi RNN untuk terjemahan bahasa

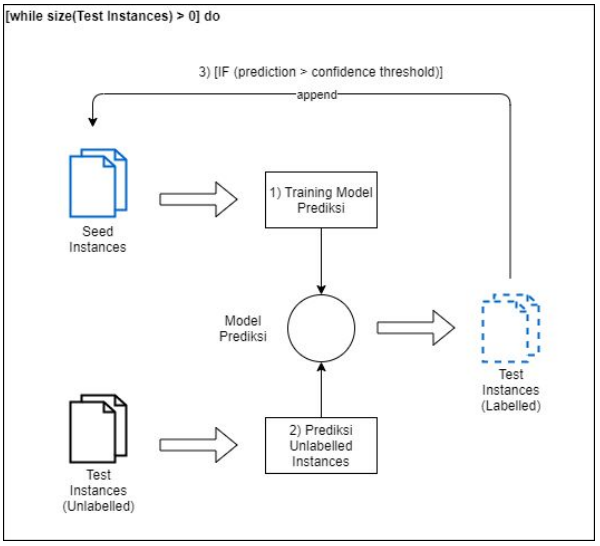
Salah satu variasi dari RNN yang paling umum adalah LSTM (*Long-Short Term Memory*). LSTM mengatasi salah satu kekurangan dari RNN, yaitu tidak mampu memahami *long-term dependencies* antar neuron. LSTM memiliki satu jalur *cell state* di mana memori informasi *dependency* tidak hanya dibawa ke neuron sebelahnya, tetapi juga dibawa ke neuron-neuron berikutnya. Kemampuan LSTM ini yang membantu memahami makna pada teks dari kata-kata yang terpisah, dan telah teruji pada beberapa tahun terakhir dengan banyaknya penggunaan arsitektur LSTM untuk klasifikasi teks. Pada penelitian ini, tim peneliti bereksperimen dengan menggunakan tiga variasi RNN, yaitu *single-layer* LSTM, *bidirectional* LSTM, dan *stacked bidirectional* LSTM, yang mana setiap variasi memiliki konfigurasi arsitektur yang berbeda.

Selain RNN, variasi lain neural network yang ada saat ini adalah CNN (*Convolutional Neural Network*). CNN ditemukan pada tahun 1988 oleh Yann Lecun. CNN bekerja dengan cara menerima teks dalam bentuk matriks sebagai masukan kemudian diproses setiap lapisannya. Setiap baris matriks merepresentasikan sebuah kata yang hendak diproses.

Sebagai pembanding, tim peneliti juga menambahkan CNN meskipun CNN beberapa tahun terakhir lebih sering digunakan untuk klasifikasi citra dibandingkan teks, tetapi tidak menutup kemungkinan dapat digunakan untuk tek karena pada dasarnya memproses data menggunakan arsitektur berlapis serta data yang dimasukkan dapat berupa representasi vektor berdimensi tinggi. Selain itu lapisan konvolusi dan lapisan *pooling* pada CNN dapat digunakan untuk memanipulasi dimensi pada data.

3.4 *Bootstrapping*

Teknik *bootstrapping* pada konteks *semi-supervised learning* pada dasarnya melakukan penambahan data yang belum dilabeli ke *training* dataset berdasarkan hasil prediksi sebelumnya. Beberapa contoh penerapan *bootstrapping* pada konteks NLP yaitu untuk memprediksi hubungan semantik antar *named entity* (Batista et al., 2015) dan *named-entity recognition* (Thenmalar et al., 2015). Pada kedua penelitian tersebut, dokumen yang telah memiliki label NER akan dijadikan sebagai *seed instance* untuk data *training* awal. Model prediksi akan di-*training* menggunakan *seed instance* untuk memprediksi label pada dokumen-dokumen lainnya. Hasil prediksi yang memiliki *confidence level* yang tinggi akan ditambahkan ke *seed instance* untuk kemudian di-*training* ulang untuk memprediksi dokumen-dokumen yang belum dilabeli. Peneliti berencana menerapkan *bootstrapping* sebagai teknik dasar dalam melakukan pendekatan *semi-supervised* yang kemudian akan dibandingkan hasilnya dengan implementasi yang lebih kompleks yaitu *Neural Graph Learning*.

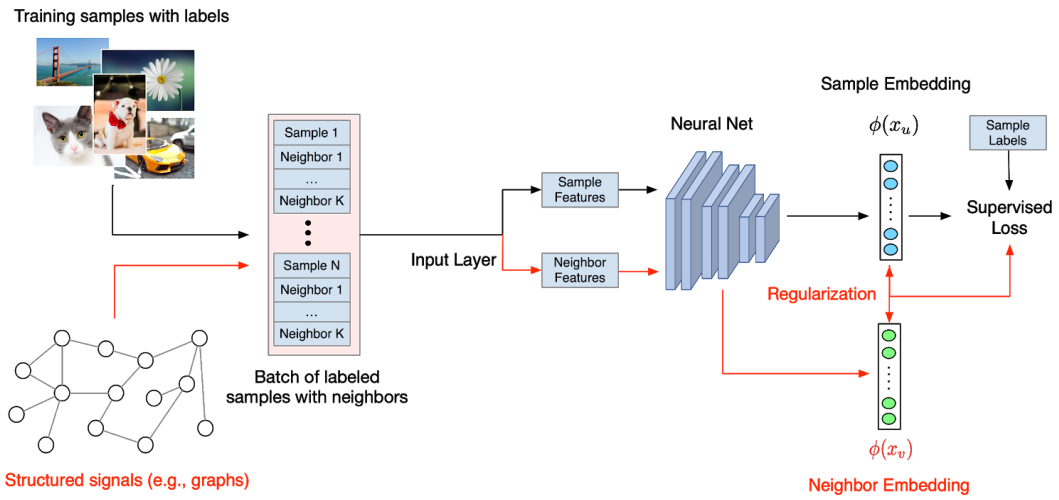


Gambar 8. Diagram alur *bootstrapping*

Pada Gambar 8, ditunjukkan cara kerja metode *bootstrapping*. Implementasi metode *bootstrapping* pada eksperimen ini yaitu pertama dengan melakukan *sampling* pada *training dataset* yang akan dijadikan sebagai *seed*. *Seed* tersebut digunakan untuk melakukan *training* model prediksi pada *dataset* sisanya. Untuk masing-masing *instance* yang diprediksi, jika probabilitas prediksi tersebut cukup tinggi (melebihi *threshold* yang ditentukan oleh peneliti), maka *instance* tersebut akan ditambahkan ke *seed* sebelumnya untuk dijadikan sebagai *training dataset* yang baru. Kemudian model akan di-*retrain* dengan *training dataset* yang baru. Proses ini akan terus berulang dan akan berakhir pada salah satu dari dua kondisi, yaitu tidak ada penambahan *training dataset* pada satu iterasi, atau telah melebihi batas iterasi yang ditentukan oleh tim peneliti.

3.5 *Neural Graph Learning*

Neural Graph Learning (NGL) adalah paradigma baru melatih *neural network* dengan memanfaatkan sinyal terstruktur sebagai masukan. Neural Graph Learning diperkenalkan pada penelitian Bui et. al (2018) dengan motivasi meningkatkan kinerja dari *neural network* dalam melakukan *training*, terutama untuk data yang memiliki tingkat kesamaan (*similarity*) tinggi antara satu data dengan data yang lain. Sinyal terstruktur digunakan untuk mewakili hubungan atau kesamaan yang ada di antara data. Data yang dimasukkan dapat berupa data berlabel maupun tidak berlabel.



Gambar 9. Alur Model *Neural Graph Learning* (NGL)

NGL menerima dua jenis input, yaitu *training sample* dengan label dan *structured signals* dalam bentuk graf. Karena belum adanya graf pada dataset, maka graf perlu disintesis dari kesamaan (*similarity*) antar *instance*, baik data berlabel maupun tidak berlabel. Setiap *vertex* didefinisikan sebagai sebuah *instance* dokumen, dan setiap *edge* didefinisikan sebagai hubungan dua arah yang melambangkan terdapat kesamaan yang tinggi berdasarkan ukuran *similarity*, dalam eksperimen ini akan digunakan L2 *similarity*. Ukuran vektor untuk masing-masing dokumen diperoleh dari hasil output *word embedding*.

Seperti pada proses *training* model *deep neural network*, terdapat *loss* yang dihasilkan pada setiap iterasi (epoch) *training*. *Loss* adalah ukuran seberapa besar ketidaksesuaian antara label prediksi dengan label yang sebenarnya. Ukuran *loss* akan dijadikan sebagai input pada proses *regularization*, yaitu proses penyesuaian parameter pada model *deep neural network*.

Pada konteks NGL, graf *similarity* akan berfungsi sebagai *regularizer* pada model prediksi *neural network* yang ada. *reg* Secara teori, implementasi ini dapat meningkatkan kualitas hasil prediksi pada kondisi minimnya data berlabel dengan memanfaatkan *similarity graph* yang ada. Teknik ini dikatakan bisa

menghasilkan akurasi yang tinggi dengan menggunakan komposisi data berlabel yang lebih sedikit daripada tidak berlabel. Pada penelitian ini, tim peneliti menggunakan *framework* yang bernama *Neural Structured Learning* dari TensorFlow yang didalamnya menyediakan *support* untuk melakukan *Neural Graph Learning* (NGL) dan *Adversarial Learning*. Pada eksperimen ini, peneliti hanya berfokus pada *Neural Graph Learning.*

3.6 Metrik Evaluasi

Setelah melakukan eksperimen, tim peneliti melakukan analisis performa model yang dirancang menggunakan empat buah metrik utama, yaitu akurasi, *precision*, *recall*, dan F1. Skor yang menjadi acuan tim peneliti adalah skor F1, karena cocok digunakan untuk kasus data yang kurang seimbang, serta menggambarkan proporsi harmonik antara nilai akurasi, *precision*, dan *recall*.

$$Akurasi = (\sum_{i=1}^{n} \frac{True\ Positive + False\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative})/n$$

Precision	Recall	F1
$(\sum_{i=1}^{n} \frac{True\ Positive}{True\ Positive + False\ Positive})/n$	$(\sum_{i=1}^{n} \frac{True\ Positive}{True\ Positive + False\ Negative})/n$	$\frac{2\ Precision \bullet Recall}{Precision + Recall}$

True Positive (TP) merupakan banyaknya data positif yang terklasifikasi dengan benar
True Negative (TN) merupakan banyaknya data negatif yang terklasifikasi dengan benar
False Positive (FP) merupakan banyaknya data negatif namun terdeteksi sebagai data positif
False Negative (FN) merupakan banyaknya data positif namun terdeteksi sebagai data negatif

BAB 4
DESAIN DAN IMPLEMENTASI PENAMBANGAN DATA

4.1 Lingkungan Pengembangan

Selama penelitian, tim peneliti menggunakan bahasa pemrograman Python dengan bantuan konsol interaktif Google Colaboratory dengan memanfaatkan *library* beserta versinya sebagai berikut:

NLTK	(3.2.5)	Tweepy	(3.6.0)
Scikit-learn	(0.22.2)	NumPy	(1.18.5)
Keras	(2.3.0)	Pandas	(1.0.5)
Tensorflow GPU	(2.4.0)	Matplotlib	(3.2.2)

Saat pelatihan data, tim pengembang menggunakan mesin virtual yang tersedia pada *Google Compute Engine* dengan spesifikasi berikut: Intel(R) Xeon(R) CPU @ 2.30GHz CPU, 32 GB ruang penyimpanan persisten, NVIDIA Tesla K80 GPU, Sistem Operasi Ubuntu 18.04 LTS, dan RAM 13 GB.

4.2 Implementasi Pendekatan *Supervised Learning*

4.2.1 Implementasi *Machine Learning* Klasik

Pada pendekatan ini, tim peneliti bereksperimen dengan 6 buah algoritma *machine learning* klasik, yaitu Multinomial Naive Bayes, Bernoulli Naive Bayes, SVM (Support Vector Machine), Decision Tree, dan KNN (K-Nearest Neighbour). Setiap algoritma menggunakan fitur ekstraksi teks yang pada subbab 3.1.

Dikarenakan penelitian merupakan klasifikasi multikelas (prediksi 5 kelas emosi), maka SVM menggunakan teknik *one-vs-rest*, dimana membandingkan satu kelas dengan kelas lainnya dalam satu waktu. KNN dipergunakan karena telah terbukti menghasilkan performa yang baik untuk klasifikasi teks sebagaimana penelitian dilakukan oleh Darujati (2010) yang mencari tahu performa klasifikasi dokumen menggunakan KNN. Tim peneliti menggunakan 10-fold *cross-validation* untuk membagi data menjadi *training* dan *validation* data.

4.2.2 Implementasi *Deep Neural Network*

Tidak hanya menggunakan algoritma *machine learning* klasik, tim peneliti juga menggunakan beberapa arsitektur *deep learning* yang akan dibandingkan performanya. Jenis pendekatan *deep learning* yang digunakan adalah *deep neural network*. Pada percobaan pertama, peneliti menggunakan salah satu variasi dari RNN, yaitu LSTM. Pemilihan arsitektur ini didasarkan atas sifat teks yang dapat membawa suatu arti atau konteks dari satu kata ke kata lainnya secara sekuensial namun tetap"mengingat" makna kata dari aliran sebelumnya.

Tim peneliti juga akan menguji beberapa variasi dari LSTM, seperti *single-layer* LSTM, *bidirectional* LSTM, dan *stacked bidirectional* LSTM. Berikut adalah spesifikasi *hyperparameter* dan arsitektur model yang tim peneliti gunakan.

`optimizer`	`= adam`	Jenis algoritma optimisasi untuk meningkatkan performa model
`max_length`	`= 80`	Panjang maksimum *sequence* kata yang untuk proses training
`trunc_type`	`= post`	Pemotongan di akhir jika ada kalimat yang panjangnya > max_length
`padding_type`	`= post`	Melakukan *padding* jika ada kalimat yang panjangnya < max_length
`oov_tok`	`= <OOV>`	Tanda khusus untuk kata yang berada di luar representasi *embedding*

Peneliti mencatat *summary* arsitektur model untuk lapisan pertama (*embedding*) berdimensi 100.

Tabel 6. *Summary* model LSTM

Layer	Output Shape	Jumlah Training Parameter
Embedding (100 dim)	(None, 80, 100)	1.661.400
LSTM (80 units)	(None, 80)	57.920
Dropout (20% rate)	(None, 80)	0
Dense (32 units)	(None, 32)	2.592
Dense (6 units)	(None, 6)	198

Tabel 7. *Summary* model Bi-LSTM

Layer	Output Shape	Jumlah Training Parameter
Embedding (100 dim)	(None, 80, 100)	1.661.400
Bidirectional LSTM (80 units)	(None, 160)	115.840
Dropout (20% rate)	(None, 160)	0
Dense (32 units)	(None, 32)	5.152
Dense (6 units)	(None, 6)	198

Tabel 8. *Summary* model Stacked Bi-LSTM

Layer	Output Shape	Jumlah Training Parameter
Embedding (100 dim)	(None, 80, 100)	1.661.400
Bidirectional LSTM (100 units)	(None, 80, 200)	160.800
Dropout (20% rate)	(None, 80, 200)	0
Bidirectional LSTM (100 units)	(None, 200)	240.800
Dropout (20% rate)	(None, 200)	0
Dense (32 units)	(None, 32)	6.432
Dense (6 units)	(None, 6)	198

Selanjutnya terdapat arsitektur *Convolutional Neural Network* (CNN). Umumnya arsitektur ini lebih banyak digunakan dalam pengolahan citra dibandingkan pengolahan teks. Arsitektur ini dapat terdiri dari beberapa *layer* konvolusi yang berfungsi untuk mentransformasikan kata-kata yang ada berdasarkan kata-kata di sampingnya. Pada eksperimen ini, pemilihan filter untuk setiap *Convolution layer* dilakukan secara otomatis oleh *library* Keras. Untuk setiap *Convolution layer*, digunakan *Max Pooling layer* untuk memperoleh hasil maksimal dari hasil filter yang di-*pooling*.

Tabel 9. *Summary* model *Convolutional Neural Network*

Layer	Output Shape	Jumlah Training Parameter
Embedding (100 dim)	(None, 80, 100)	1.651.300
Convolution 1D (Kernel = 5)	(None, 76, 64)	32.064
Max Pooling 1D	(None, 38, 64)	0
Convolution 1D (Kernel = 5)	(None, 34, 32)	10.272
Global Average Pooling 1D	(None, 32)	0
Dropout (20% rate)	(None, 32)	0
Dense (32 units)	(None, 32)	1.056
Dense (6 units)	(None, 6)	198

Terakhir, tim peneliti mengajukan arsitektur yang sederhana, yaitu arsitektur yang hanya berbasis pada *word embedding*. Alasan tim peneliti menggunakan arsitektur ini adalah akibat bentuk dari teks media sosial yang cenderung tidak formal dan tidak terstruktur, sehingga risiko untuk *overfitting* pada saat proses *training* pada arsitektur-arsitektur yang lebih kompleks menjadi lebih tinggi. Terdapat dua variasi dari arsitektur jenis ini, pertama yaitu dengan menerapkan *average pooling layer* untuk menurunkan dimensi vektor dari hasil *embedding*. Variasi kedua yaitu dengan menerapkan *flattening* pada hasil *embedding*.

Tabel 10. *Summary* model Embedding + Pooling

Layer	Output Shape	Jumlah Training Parameter
Embedding (100 dim)	(None, 80, 100)	1.661.400
Global Average Pooling 1D	(None, 100)	0
Dropout (20% rate)	(None, 100)	0
Dense (32 units)	(None, 32)	3.232
Dense (6 units)	(None, 6)	198

Tabel 11. *Summary* model Embedding + Flatten

Layer	Output Shape	Jumlah Training Parameter
Embedding (100 dim)	(None, 80, 100)	1.651.300
Flatten	(None, 8.000)	0
Dense (64 units)	(None, 64)	512.064
Dense (32 units)	(None, 32)	2.080
Dense (6 units)	(None, 6)	198

Secara total, dari ketiga jenis arsitektur tersebut, terdapat enam arsitektur *deep learning* yang akan dibandingkan performanya dalam memprediksi emosi yang terindikasi pada teks *tweet*. Tim peneliti telah mencakup arsitektur *deep learning* dari berbagai skala kompleksitas, mulai dari arsitektur yang sederhana (hanya *word embedding*) hingga yang kompleks (LSTM, CNN).

4.3 Implementasi Pendekatan *Semi-Supervised Learning*

4.3.1 Implementasi *Bootstrapping*

Pada eksperimen ini, tim peneliti mencoba variasi dari nilai *confidence threshold* mulai dari 0.95, 0.975 dan 0.99 dengan maksimum iterasi sebanyak 50 kali. Algoritma prediksi yang digunakan yaitu algoritma Embedding + Pooling karena algoritma tersebut yang memperoleh hasil terbaik dalam skenario *supervised learning* yang akan dibahas pada bab berikutnya. Beberapa skenario *supervision rate* juga akan diuji, mulai dari 10%, 5%, hingga 2%. *Supervision rate* tersebut mendefinisikan rasio data yang telah dilabeli sebagai input oleh pengguna (*seed*) yang akan digunakan untuk memprediksi dan melabeli data lainnya.

4.3.2 Implementasi *Neural Graph Learning*

Pada eksperimen ini, tim peneliti mencoba variasi dari nilai minimum *similarity* untuk sintesis graf mulai dari 0.95, 0.975 dan 0.99 dengan maksimum *neighbors* sebanyak 3 *instance*. Algoritma prediksi yang digunakan yaitu algoritma Embedding + Pooling karena algoritma tersebut yang memperoleh hasil terbaik dalam skenario *supervised learning* yang akan dibahas pada bab berikutnya. Beberapa skenario *supervision rate* juga akan diuji, mulai dari 10%, 5%, hingga 2%. *Supervision rate* tersebut mendefinisikan rasio perbandingan antara data yang telah dilabeli sebagai input oleh pengguna dengan keseluruhan data yang ada dan dijadikan sebagai input untuk sintesis *similarity graph*.

BAB 5
HASIL PENELITIAN DAN ANALISIS

5.1 Hasil Pendekatan *Supervised Learning*

Pada percobaan awal, tim peneliti menggunakan enam jenis algoritma *machine learning* klasik, dengan menggunakan teknik 10-fold K-fold *cross-validation* untuk membagi data menjadi *training* dan *validation*, serta masing - masing diuji menggunakan empat jenis ekstraksi teks (BoW, Leksikon, POS Tagging, dan Ortografi) sebagai fitur pada data. Hasil terbaik diambil berdasarkan nilai F1 yang tertinggi untuk setiap percobaan yang dilakukan. Berikut adalah tabel hasil pengujian.

Tabel 12. Hasil eksperimen *supervised learning* dengan *machine learning* klasik

Fitur	Multinomial Naive Bayes			
	Akurasi	Precision	Recall	F1
BoW	**0,5491**	**0.5408**	**0.5479**	**0.5358**
Leksikon	0,3216	0.2051	0.2549	0.1709
POS	0,2547	0.1015	0.2012	0.1187
Ortografi	0,2816	0.1988	0.2413	0.1828

Fitur	Logistic Regression			
	Akurasi	Precision	Recall	F1
BoW	**0,5441**	**0.5591**	**0.5394**	**0.5399**
Leksikon	0,3248	0.2691	0.2715	0.2107
POS	0,2589	0.1704	0.2073	0.1437
Ortografi	0,2952	0.1791	0.2429	0.2019

Fitur	Bernoulli Naive Bayes			
	Akurasi	Precision	Recall	F1
BoW	**0,5519**	**0.5505**	**0.5472**	**0.5371**
Leksikon	0,3243	0.1817	0.2609	0.2056
POS	0,2595	0.1039	0.2056	0.1373
Ortografi	0,2668	0.1763	0.2159	0.1487

Fitur	K-Nearest Neighbour (k=3)			
	Akurasi	Precision	Recall	F1
BoW	**0,3477**	**0.3774**	**0.3268**	**0.3259**
Leksikon	0,2538	0.2459	0.2249	0.1818
POS	0,2425	0.1655	0.2011	0.1255
Ortografi	0,2551	0.2304	0.2285	0.2167

Fitur	Support Vector Machine (*one-vs-rest*)			
	Akurasi	Precision	Recall	F1
BoW	**0,5055**	**0.5044**	**0.5055**	**0.5008**
Leksikon	0,3233	0.2055	0.2697	0.2038
POS	0,2587	0.1754	0.2064	0.141
Ortografi	0,2031	0.1098	0.2053	0.0889

Fitur	Decision Tree			
	Akurasi	Precision	Recall	F1
BoW	**0,4549**	**0.4622**	**0.4613**	**0.4562**
Leksikon	0,3236	0.2881	0.2725	0.2354
POS	0,2525	0.1172	0.2007	0.1344
Ortografi	0,2375	0.2213	0.2211	0.2205

Hasil terbaik diperoleh untuk fitur BoW untuk semua percobaan dengan rerata skor diatas 50% kecuali untuk algoritma KNN dan Decision Tree. Peneliti menduga hal ini disebabkan karena fitur BoW berisi 2000 kata unik dengan frekuensi kemunculan terbanyak pada data, kemudian distribusi 2000 kata tersebut cukup seimbang dan muncul di banyak *tweet*, sehingga menghasilkan nilai akurasi, *precision*, *recall*, dan F1 yang stabil. Hasil terburuk diperoleh oleh fitur POS Tagging. Peneliti menduga hal ini disebabkan karena POS Tagging yang digunakan terbentuk dari kata - kata dasar kamus Bahasa Indonesia, sedangkan data yang digunakan oleh peneliti tidak melalui prosedur *stemming*, sehingga setiap kata pada data tidak berbentuk kata dasar kamus serta cenderung tidak baku. Hal ini menyebabkan performa prediksi rendah.

Berikutnya, tim peneliti menguji arsitektur *deep neural network* dengan pembagian dataset 60/40 untuk *training* dan *validation*. Secara total terdapat enam arsitektur yang akan diuji, dan empat variasi nilai dimensi *embedding*, yaitu 50, 100, 200, dan 300. Berikut adalah tabel hasil pengujian.

Tabel 13. Hasil eksperimen *supervised learning* dengan arsitektur *deep neural network*

Dimensi Embed	LSTM			
	Akurasi	Precision	Recall	F1
50	0,2	0.0480	0.2000	0.0774
100	0,2	0.0480	0.2000	0.0774
200	0,2	0.0480	0.2000	0.0774
300	**0,2**	**0.0480**	**0.2000**	**0.0774**

Dimensi Embed	Single Layer Bidirectional LSTM			
	Akurasi	Precision	Recall	F1
50	**0,5038**	**0.5350**	**0.5044**	**0.5127**
100	0,5045	0.5151	0.5038	0.5073
200	0,5011	0.5208	0.4919	0.5004
300	0,4917	0.5160	0.4900	0.4990

Dimensi Embed	Stacked Bidirectional LSTM			
	Akurasi	Precision	Recall	F1
50	0,3367	0.4063	0.3527	0.3376
100	**0,4612**	**0.4877**	**0.4597**	**0.4699**
200	0,3856	0.3984	0.3872	0.3882
300	0,3774	0.4106	0.3815	0.3812

Dimensi Embed	Convolutional Neural Network			
	Akurasi	Precision	Recall	F1
50	0,4323	0.4405	0.4377	0.4346
100	0,4327	0.4702	0.4333	0.4421
200	0,4586	0.4771	0.4591	0.4621
300	**0,4786**	**0.4883**	**0.4813**	**0.4739**

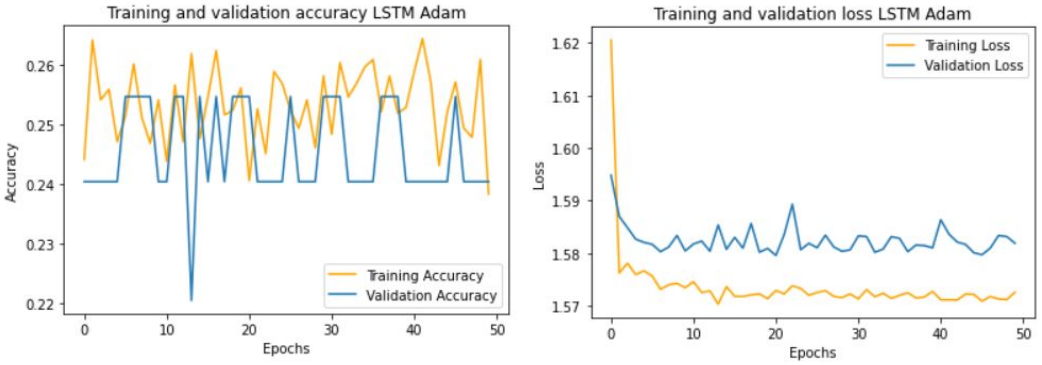
Dimensi Embed	Embedding + Pooling			
	Akurasi	Precision	Recall	F1
50	**0,5530**	**0.5848**	**0.5471**	**0.5606**
100	0,5482	0.5630	0.5455	0.5514
200	0,5233	0.5399	0.5201	0.5268
300	0,5455	0.5542	0.5434	0.5439

Dimensi Embed	Embedding + Flatten			
	Akurasi	Precision	Recall	F1
50	0,4605	0.5317	0.4579	0.4719
100	**0,4804**	**0.4820**	**0.4819**	**0.4768**
200	0,45	0.4576	0.4652	0.4527
300	0,4733	0.4787	0.4733	0.4754

Hasil terbaik berdasarkan nilai akurasi dan skor F1 diraih oleh model Embedding + Pooling dengan parameter jumlah dimensi *embedding* sebesar 50. Walaupun begitu, semua nilai dari model Embedding + Pooling masih mengungguli model lainnya. Hasil terbaik kedua diraih oleh arsitektur *single-layer Bidirectional* LSTM dengan nilai akurasi dan skor F1 di kisaran 50%. Secara umum, *embedding* dengan dimensi sebesar 100 menghasilkan skor terbaik, kecuali untuk model *Convolutional Neural Network*, yang skornya masih meningkat bahkan pada parameter 300 dimensi *word embedding*. Peneliti menduga adanya kasus *overfitting* dengan *word embedding* lebih dari 100 dimensi, karena adanya penambahan fitur/dimensi, sehingga gagal menggeneralisasi dataset. Di sisi lain, diduga terjadi kasus *underfit* pada dimensi *word embedding* di bawah 100 dimensi, di mana jumlah fitur (dimensi *word embedding*) yang digunakan kurang banyak sehingga kurang mampu untuk membedakan masing-masing kata ketimbang dengan 100 dimensi *word embedding*.

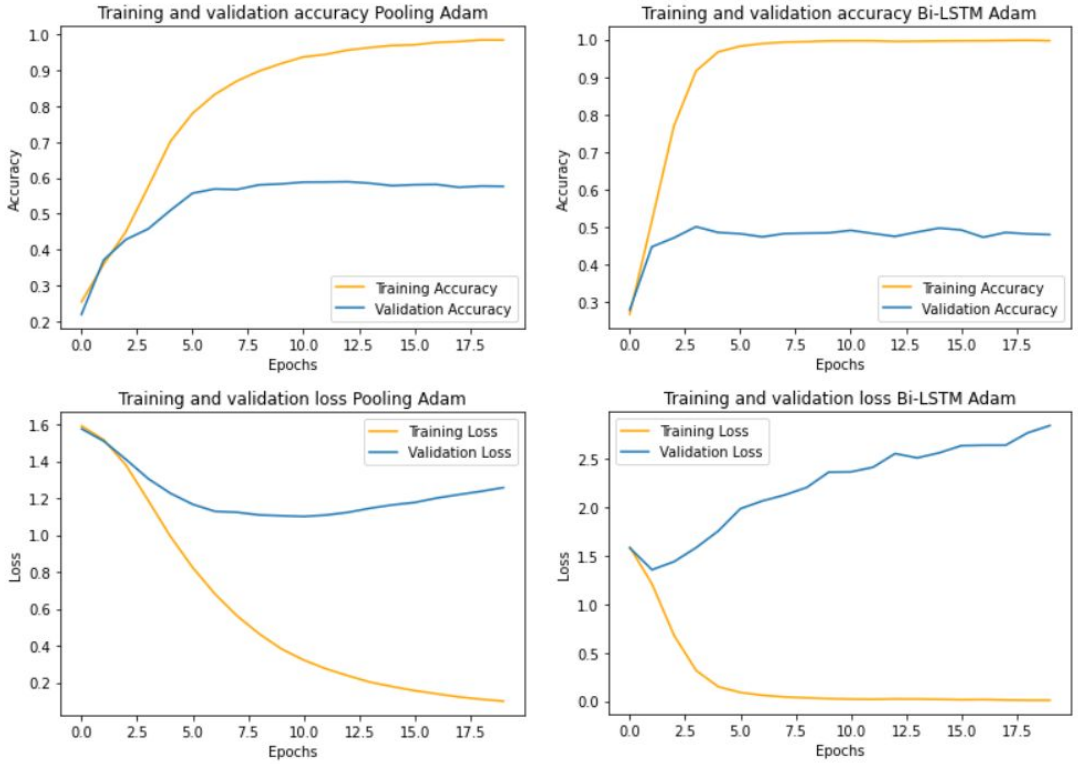
Selanjutnya terdapat beberapa kasus yang menurut tim peneliti menarik untuk dibahas yang juga disajikan visualisasi grafnya. Hal pertama yang perlu dibahas oleh peneliti yaitu gagalnya arsitektur LSTM untuk memprediksi data yang dimiliki tim peneliti. Peneliti telah mencoba meningkatkan jumlah epoch hingga 50, namun tidak membuahkan hasil yang memuaskan. Seperti yang ditunjukkan pada Gambar 10, nilai akurasi tidak mengalami kenaikan selama proses *training* dan hanya berkisar di antara 24% hingga 26%. Gambar

tersebut dihasilkan pada parameter 100 dimensi *word embedding*, tetapi nilai akhir yang diperoleh untuk setiap variasi dimensi *word embedding* tetaplah sama. Peneliti menduga hal ini terjadi dikarenakan arsitektur yang digunakan lebih sederhana dibandingkan variasi LSTM lainnya, sehingga harus ditambahkan lapisan lain pada arsitektur LSTM.



Gambar 10. Graf nilai akurasi dan *loss* pada *training* model LSTM

Pada dua model arsitektur dengan skor tertinggi, Embedding + Pooling dan Bi-LSTM, nilai akurasi dan skor F1 dapat melebihi 50% dengan parameter 100 dimensi *word embedding* seperti pada Gambar 11. Namun berdasarkan nilai akurasi dan *loss* selama proses *training*, peneliti dapat menyimpulkan bahwa model Embedding + Pooling menghasilkan nilai yang lebih stabil selama proses *training*, seperti yang ditunjukkan pada dibawah. Model Embedding + Pooling membutuhkan lebih banyak *epoch* untuk memperoleh hasil optimal dibandingkan dengan model Bi-LSTM. Jika dibandingkan dengan prediksi algoritma *machine learning* klasik, model Embedding + Pooling sedikit unggul dibandingkan dengan algoritma Bernoulli Naive Bayes pada fitur BoW dengan selisih akurasi sebesar 0,11% dan selisih skor F1 sebesar 2,35%.



Gambar 11. Graf nilai akurasi dan *loss* pada *training* model Embedding + Pooling (kiri) dan Bi-LSTM (kanan)

5.2 Hasil Pendekatan *Semi-Supervised Learning*

Berdasarkan hasil perbandingan model prediksi pada skenario *supervised learning*, peneliti memilih model Embedding + Pooling untuk digunakan pada skenario *semi-supervised learning*. Parameter yang diatur oleh peneliti yaitu *confidence threshold* untuk *bootstrapping* atau *similarity threshold* untuk NGL, dan *supervision rate*. Dataset dibagi menjadi tiga bagian, yaitu *train* dan *test* dengan rasio 50/50. Dataset *train* kemudian dibagi lagi menjadi dua, yaitu data berlabel dan data tidak berlabel dengan rasio sesuai dengan *supervision rate*.

Berdasarkan hasil eksperimen pada tabel dibawah, performa NGL lebih unggul secara keseluruhan dibandingkan dengan performa *bootstrapping*. Pada semua hasilnya, peneliti merasa nilai F1 *score* bisa terbilang cukup seimbang dengan nilai *precision* dan *recall*, walaupun nilai *precision* sedikit lebih tinggi, atau jauh lebih tinggi pada beberapa skenario yang buruk (*supervision rate* rendah, *threshold* tinggi). Hasil terbaik diambil berdasarkan nilai F1 tertinggi dari setiap percobaan yang dilakukan peneliti.

Tabel 14. Hasil eksperimen *semi-supervised learning*

Supervision Rate	Bootstrapping (Confidence Threshold=0.95)			
	Akurasi	Precision	Recall	F1
0.10	**0,3685**	**0.4514**	**0.3444**	**0.3354**
0.05	0,3037	0.4019	0.2821	0.2829
0.02*	0,2745	0.3205	0.2999	0.2442

Supervision Rate	NGL (Similarity Threshold=0.95)			
	Akurasi	Precision	Recall	F1
0.10	0,5526	0.5649	0.5450	0.5491
0.05	**0,5490**	**0.5896****	**0.5344**	**0.5513**
0.02	0,5530	0.5767	0.5282	0.5400

Supervision Rate	Bootstrapping (Confidence Threshold=0.975)			
	Akurasi	Precision	Recall	F1
0.10	**0,3937**	**0.4651**	**0.3754**	**0.3802**
0.05	0,2697	0.2718	0.2170	0.1246
0.02*	0,2514	0.3170	0.2465	0.2107

Supervision Rate	NGL (Similarity Threshold=0.975)			
	Akurasi	Precision	Recall	F1
0.10	**0,5458**	**0.5670**	**0.5388**	**0.5498**
0.05	0,5485	0.5705	0.5346	0.5450
0.02	0,5172	0.6081**	0.4873	0.5058

Supervision Rate	Bootstrapping (Confidence Threshold=0.99)			
	Akurasi	Precision	Recall	F1
0.10	**0,3907**	**0.4674**	**0.3861**	**0.3846**
0.05*	0,3467	0.3942	0.3646	0.3500
0.02*	0,2511	0.2992	0.2981	0.2354

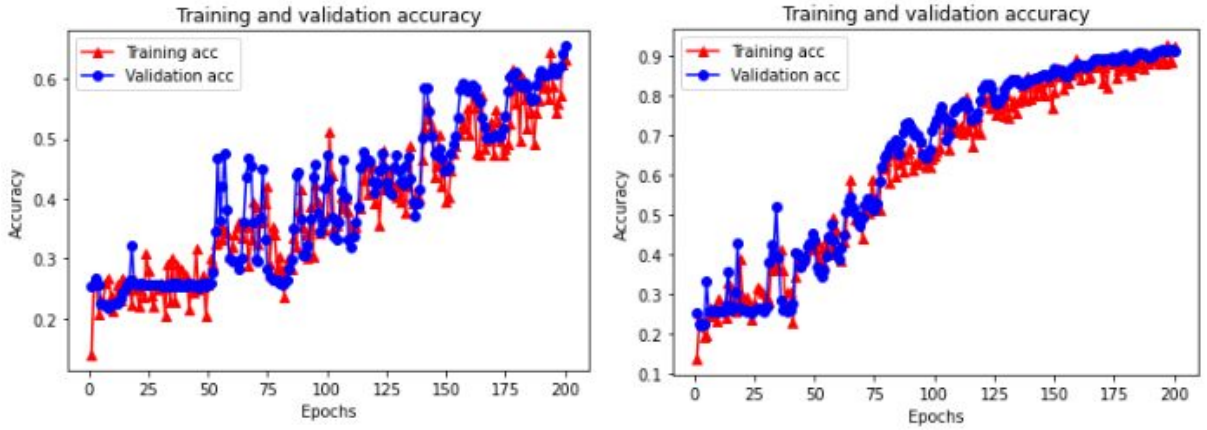
Supervision Rate	NGL (Similarity Threshold=0.99)			
	Akurasi	Precision	Recall	F1
0.10	**0,5644****	**0.5894****	**0.5549****	**0.5667****
0.05	0,5647**	0.6015**	0.5498**	0.5657**
0.02	0,5156	0.5848	0.4764	0.4886

Keterangan tabel:
*: (khusus untuk bootstrapping) Penambahan data pada data training oleh prediksi kurang dari 5%
**: Measurement melebihi hasil terbaik dari algoritma Embedding + Pooling di skenario *supervised learning*

Performa *bootstrapping* cenderung memburuk seiring dengan peningkatan parameter *confidence threshold*. Pada skenario yang buruk, *bootstrapping* gagal menganotasi data tidak berlabel lebih dari 5% dari keseluruhan data tidak berlabel, sehingga model prediksi gagal generalisasi pada *test* data. Selain itu, dengan parameter *supervision rate* 2%, semua kondisi *threshold* gagal mengekspansi *training* data secara signifikan.

Di sisi lain, NGL dapat menyamai bahkan dalam beberapa kasus melebihi performa dari *supervised learning* dengan model yang sama. Pada kondisi *supervision rate* yang masih cukup tinggi (10% dan 5%), performa terbaik diperoleh pada *similarity threshold* yang tinggi pula (0.99). Namun, ketika *supervision rate* rendah (2%), performa terbaik diperoleh dengan parameter *graph similarity threshold* yang lebih rendah (95%). Peneliti menduga dengan *similarity threshold* yang lebih rendah, model prediksi dapat menggeneralisasi dataset dengan lebih baik.

Penggunaan *similarity graph* sebagai *regularizer* pada NGL sangat mendukung dalam proses *training* model khususnya pada kondisi *supervision rate* yang rendah. Hal ini tampak pada stabilitas dan peningkatan akurasi pada data *training* (berlabel) dan *validation* (tidak berlabel) seiring dengan bertambahnya *training epoch*. Graf perbandingan dua kondisi tersebut dapat dilihat pada gambar dibawah. Graf tersebut diperoleh dengan parameter *supervision rate* 2% dan *similarity threshold* 95%.



Gambar 12. Perbandingan performa *training* pada *non-regularized* (kiri) dengan *regularized* (kanan)

BAB 6
PENUTUP

6.1 Kesimpulan Penelitian

Berdasarkan hasil pengujian, pada skenario *supervised learning*, algoritma *deep neural network* khususnya model Embedding + Pooling mengungguli tipis algoritma-algoritma *machine learning* klasik yang terbaik, yaitu Bernoulli Naive Bayes dengan selisih akurasi sebesar 0,11% dan skor F1 sebesar 2,35%. Model Embedding + Pooling berikutnya digunakan pada skenario *semi-supervised learning*. Berdasarkan eksperimen, NGL (*Neural Graph Learning*) menghasilkan skor akurasi dan skor F1 yang lebih tinggi secara signifikan dibandingkan *bootstrapping* dengan selisih akurasi sebesar 17% dan skor F1 sebesar 18%. Hasil akhir terbaik dari NGL adalah performa yang dihasilkan dengan akurasi 56,44% dan skor F1 56,67% pada *similarity threshold* 99% dan *supervision rate* 10%.

Melalui eksperimen ini, tim peneliti telah menguji pemanfaatan *predictive analytics* menggunakan *machine learning*, khususnya *deep learning* ditambah dengan pendekatan *semi-supervised learning* sebagai sebuah solusi bagi pemerintah dan instansi pendidikan untuk memperoleh *insight* terkait *feedback* oleh masyarakat umum terhadap pendidikan di masa pandemi dalam bentuk prediksi emosi dari data *tweet*. Pemanfaatan teknologi *deep neural network* terbukti membantu meningkatkan performa dari prediksi terhadap dataset emosi pada *tweet*. Selain itu, penggunaan pendekatan *semi-supervised* terbukti *feasible* tanpa mempengaruhi performa algoritma pembelajaran mesin secara signifikan, sehingga bisa digunakan untuk mengatasi masalah kurangnya data berlabel, seperti pada kasus eksperimen ini di mana *annotator* tidak dapat melakukan anotasi data secepat arus *tweet* yang masuk.

6.2 Saran untuk Pengembangan Selanjutnya

Tim peneliti merasa pada aspek dari kualitas anotasi data dapat ditingkatkan ke depannya, seperti memanfaatkan koefisien Cohen's Kappa, untuk mengukur tingkat reliabilitas antar *annotator* untuk data kategorikal. Secara teori, dengan meningkatkan reliabilitas anotasi antar *annotator*, algoritma pembelajaran mesin yang digunakan dapat lebih mudah memahami pola yang terdapat pada masing-masing label dan berdampak pada peningkatan skor metrik evaluasi. Selain itu, dari aspek prediksi, tim peneliti merasa peningkatan hasil dapat dilakukan dengan bereksperimen lebih lanjut pada *hyperparameter tuning* baik di algoritma *machine learning* maupun pada pendekatan yang digunakan pada *semi-supervised learning*.

Melalui pengembangan lebih lanjut pada skala yang lebih besar, tim peneliti berharap pemanfaatan *predictive analytics* dapat terwujud dan diimplementasikan untuk mendukung pemerintah dan instansi pendidikan dalam pengambilan keputusan dan penentuan kebijakan terkait pelaksanaan pendidikan di masa pandemi. Pengenalan teknologi ini dapat mendukung visi menuju Indonesia Maju pada aspek pendidikan dan mewujudkan keberhasilan pelaksanaan program pendidikan.

DAFTAR PUSTAKA

Batista, D. S., Martins, B., & Silva, M. J. (2015, September). *Semi-supervised bootstrapping of relationship extractors with distributional semantics*. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (pp. 499-504).

Bui, Thang D, S. Ravi, & V. Ramavajjala. (2018). *Neural Graph Learning: Training Neural Networks Using Graphs*. Proceedings of 11th ACM International Conference on Web Search and Data Mining (WSDM) (2018).

Cahyo, D. (2010). *Perbandingan Klasifikasi Dokumen Teks Menggunakan Metode Naive Bayes dengan K-Nearest Neighbor*. Universitas Narotama.

Dinakaramani. A, F. Rashel, A.Luthfi, & R. Manurung. (2014). *Designing an Indonesian Part of speech Tagset and Manually Tagged Indonesian Corpus*. International Conference on Asian Language Processing (IALP) 2014.

Herzig, J., Shmueli-Scheuer, M., & Konopnicki, D. (2017). *Emotion Detection from Text via Ensemble Classification Using Word Embeddings*. In Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval (pp. 269-272).

Leow, G. (2019). *Scraping Tweets with Tweepy Python*. 4 September 2020. https://medium.com/@leo wg-riffin/scraping-tweets-with-tweepy-python-59413046e788.

Saputri, M. Silvana, R. Mahendra, & M. Adriani. (2018) *Emotion Classification on Indonesian Twitter Dataset*. International Conference on Asian Language Processing (IALP) 2018.

Shaver, P. R., Murdaya, U., & Fraley, R. C. (2001). *Structure of the Indonesian emotion lexicon*. Asian journal of social psychology, 4(3), 201-224.

Thenmalar, S., Balaji, J., & Geetha, T. V. (2015). *Semi-Supervised Bootstrapping Approach for Named Entity Recognition*. arXiv preprint arXiv:1511.06833.

Vania. C, Moh. Ibrahim, & Adriani, M. (2014). *Sentiment Lexicon Generation for an Under-Resourced Language*. International Journal of Computational Linguistics and Applications (IJCLA) 2014.

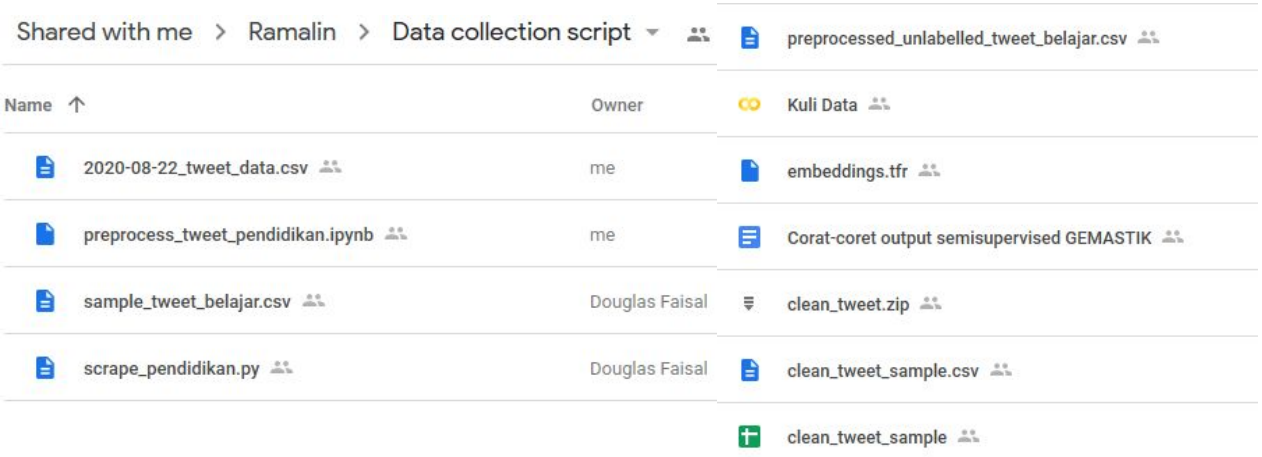
Vora, P., Khara, M., & Kelkar, K. (2017). *Classification of Tweets Based on Emotions Using Word Embedding and Random Forest Classifiers*. International Journal of Computer Applications, 178(3), 1-7.

W. Jenq-Haur, L. Ting-Wei, L. Xiong, & W. Long. (2018). *An LSTM Approach to Short Text Sentiment Classification with Word Embeddings*. The 2018 Conference on Computational Linguistics and Speech Processing (ROCLING) 2018.

DOKUMENTASI PENELITIAN

	A	B
1	label	tweet
2	sadness	<USERNAME> buat kuliah mzkun, mau beli spatu lagi malu mintanya sm ortu
3	anger	<USERNAME> Yang lain: siswa tersebut caper dan tidak bertanggung jawab atas kematiannya sendiri
4	love	Saking rindu nya gua sama teman2 Kuliah. Babi lewat pun gua kira mereka
5	sadness	Nangis di kampus soalnya belom pulang😢😢😢😢😢😢😢 <URL>
6	sadness	punten selingan gw capek belajar asl 😭 <URL>
8	sadness	Ternyata jadi guru ga semudah yg aku bayangkan gess:') <URL>
9	happy	<USERNAME> BENTAR LAGI CHAT DOSEN. DEGDEGAN.
11	sadness	Gusti males banget belajar buat ujikom😭😭
15	anger	<USERNAME> Buat yang mempermasalahin soal pengucapan alhamdulillah, coba belajar bahasa asingnya d
17	anger	kuliah di us tuh semahaaaaal itu ya, jadi suka gila aja liat orang indo bisa kuliah di sana tanpa scholarship apal
21	love	Pengen bgt ga si klen dulu yg dianggep paling bego,lemot mapel penting yg dulu nya jd bahan ketawaan banya

Gambar 13. Potongan data yang sudah dianotasi oleh tim peneliti



Gambar 14. Potongan *workspace* tim peneliti menggunakan Google Drive

```
#=============================================#
# Ramalin - Universitas Indonesia
#=============================================#
model_bi_lstm = tf.keras.Sequential([
    tf.keras.layers.Embedding(vocab_size+1, embedding_dim, input_length=train_padded.shape[1]),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(max_length)),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32, activation='sigmoid'),
    tf.keras.layers.Dense(num_labels+1, activation='softmax')
])
model_stacked_bi_lstm = tf.keras.Sequential([
    tf.keras.layers.Embedding(vocab_size+1, embedding_dim, input_length=train_padded.shape[1]),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(embedding_dim, return_sequences=True)),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(embedding_dim)),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32, activation='sigmoid',
                          kernel_regularizer=regularizers.l1_l2(l1=1e-3, l2=1e-3),
                          bias_regularizer=regularizers.l2(1e-3),
                          activity_regularizer=regularizers.l2(1e-3)),
    tf.keras.layers.Dense(num_labels+1, activation='softmax')
])
```

Gambar 15. Eksperimen perancangan model Bi-LSTM & *Stacked* Bi-LSTM dengan *library* Keras

```
compile_model(model_bi_lstm, optimizer_adam, 'Bi-LSTM Adam')
```

```
Bi-LSTM Adam
Epoch 1/20
125/125 - 7s - loss: 1.5903 - accuracy: 0.2664 - val_loss: 1.5829 - val_accuracy: 0.2777
Epoch 2/20
125/125 - 6s - loss: 1.2120 - accuracy: 0.5148 - val_loss: 1.3580 - val_accuracy: 0.4462
Epoch 3/20
125/125 - 6s - loss: 0.6800 - accuracy: 0.7689 - val_loss: 1.4418 - val_accuracy: 0.4703
Epoch 4/20
125/125 - 7s - loss: 0.3168 - accuracy: 0.9157 - val_loss: 1.5866 - val_accuracy: 0.5000
Epoch 5/20
125/125 - 7s - loss: 0.1494 - accuracy: 0.9666 - val_loss: 1.7585 - val_accuracy: 0.4846
```

Gambar 16. Proses *training* satu dari enam model *deep learning- supervised learning*

```
def generate_prediction_result(clf):
    feat_list = [unigram_feat, sentlex_feat, postag_feat, orto_feat]
    feat_name = ["Unigram", "Sentimen", "POS", "Ortografi"]
    for f, n in zip(feat_list, feat_name):
        X = f
        y = label
        # Mendapatkan skor selain confusion matrix
        scoring = ['accuracy', 'precision_macro', 'recall_macro', 'f1_macro']
        scores = cross_validate(clf, X, y, cv=10, scoring=scoring)
        print("Jenis Fitur : ", n)
        print("======================")
        print("Akurasi :", np.mean(scores['test_accuracy']))
        print("Macro-Precision :", np.mean(scores['test_precision_macro']))
        print("Macro-Recall :", np.mean(scores['test_recall_macro']))
        print("Macro-F1 :", np.mean(scores['test_f1_macro']))
        print("======================")
```

Gambar 17. Potongan kode evaluasi model *machine learning* klasik

```
#============================================#
# Ramalin - Universitas Indonesia
#============================================#

======== Iter 0 ========
100%|[redacted]| 3255/3255 [01:41<00:00, 31.93it/s]
104/104 [==============================] - 0s 2ms/step - loss: 1.6735 - accuracy: 0.2797
Result:  [1.6735109090805054, 0.27965080738067627]
precision: 0.3467771460840902
recall: 0.3088129417910812
fscore: 0.2299707289936673
Added 1 new labelled data at iter 0
======== Iter 1 ========
100%|[redacted]| 3254/3254 [01:42<00:00, 31.60it/s]
104/104 [==============================] - 0s 2ms/step - loss: 1.6375 - accuracy: 0.2511
Result:  [1.6375048160552979, 0.2510535717010498]
precision: 0.29915658946491175
recall: 0.298117920477713
fscore: 0.23535624651703926
No new labelled data found, break iteration
```

Gambar 18. Proses *training* menggunakan pendekatan *Bootstrapping - semi-supervised learning*

```
#================================================#
# Ramalin - Universitas Indonesia
#================================================#
graph_builder_config = nsl.configs.GraphBuilderConfig(
    similarity_threshold=0.85, lsh_splits=32, lsh_rounds=15, random_seed=12345)
nsl.tools.build_graph_from_config(['/tmp/dataset/embeddings.tfr'],
                                  '/tmp/dataset/graph_85.tsv',
                                  graph_builder_config)
graph_builder_config = nsl.configs.GraphBuilderConfig(
    similarity_threshold=0.80, lsh_splits=32, lsh_rounds=15, random_seed=12345)
nsl.tools.build_graph_from_config(['/tmp/dataset/embeddings.tfr'],
                                  '/tmp/dataset/graph_80.tsv',
                                  graph_builder_config)
```

Gambar 19. Eksperimen Pembentukan graf pada *Neural Graph Learning*