# NO: 20011538/KOM/2021

**IMPLEMENTASI WEB SCRAPING DAN KLASIFIKASI DALAM PEMBUATAN PORTAL BERITA ONLINE**

**SKRIPSI**

Digunakan untuk memenuhi persyaratan penyelesaian program S-1

Program Studi Informatika Fakultas Teknologi Industri

Universitas Kristen Petra

Oleh :

Jehezkiel Hardwin Tandijaya

NRP : C14170054

**PROGRAM STUDI INFORMATIKA**

****

**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS KRISTEN PETRA**

**SURABAYA**

**2021**

# LEMBAR PENGESAHAN

**SKRIPSI**

**IMPLEMENTASI WEB SCRAPING DAN KLASIFIKASI DALAM PEMBUATAN PORTAL BERITA ONLINE**

Oleh :

Jehezkiel Hardwin Tandijaya NRP : C14170054

Diterima Oleh :

Program Studi Informatika

Fakultas Teknologi Industri

Universitas Kristen Petra

Surabaya, 13 Juli 2020

|  |  |  |
| --- | --- | --- |
| Pembimbing I |  | Pembimbing II |
|  |  |  |
|  |  |  |
| LILIANA, S.T., M.Eng., Ph.D.  NIP: 03024 |  | Dr.Ing. INDAR SUGIARTO, S.T., M.Sc.  NIP: 02002 |

Ketua Tim Penguji

HENRY NOVIANUS PALIT, S.Kom., M.Kom., Ph.D.

NIP: 14001

Kepala Program Studi

HENRY NOVIANUS PALIT, S.Kom., M.Kom., Ph.D.

NIP: 14001

# LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai mahasiswa Universitas Kristen Petra, yang bertanda tangan dibawah ini, saya:

Nama : Jehezkiel Hardwin Tandijaya

NRP : C14170054

Demi mengembangkan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Kristen Petra Hak Bebas Royalti Non-Eksklusif (*Non- Exclusive Royalti-Free Rights*) atas karya ilmiah saya yang berjudul: Implementasi Web Scraping dan Klasifikasi dalam Pembuatan Portal Berita Online. Dengan Hak Bebas Royalti Non-Eksklusif ini Universitas Kristen Petra berhak menyimpan, mengalih-media format-kan, mengelolanya dalam bentuk pangkalan data (*database*), mendistribusikannya dan menampilkan/ mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa perlu meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Universitas Kristen Petra, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

|  |  |
| --- | --- |
|  | Surabaya, 13 Juni 2021 |
|  | Yang menyatakan, |
|  |  |
|  |  |
|  | (Jehezkiel Hardwin Tandijaya) |

# KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas berkat dan pimpinan-Nya yang telah diberikan selama pengerjaan skripsi ini, sehingga penulis dapat menyelesaikan skripsi ini dengan baik.

Pada kesempatan ini penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada orang-orang yang telah berperan sehingga skripsi ini dapat terselesaikan, antara lain:

1. Henry Novianus Palit, Ph.D., selaku Ketua Program Studi Teknik Informatika dan Sistem Informasi Bisnis Universitas Kristen Petra.
2. Liliana, S.T., M.Eng., Ph.D., selaku dosen pembimbing I, yang telah memberikan arahan, motivasi serta meluangkan waktu selama proses pembuatan skripsi berlangsung.
3. Dr.Ing. INDAR SUGIARTO, S.T., M.Sc., selaku dosen pembimbing II, yang telah memberikan arahan, motivasi serta meluangkan waktu selama proses pembuatan skripsi berlangsung.
4. Silvia Rostianingsih, M.MT., selaku Koordinator Skripsi Program Teknik Informatika dan Sistem Informasi Bisnis Universitas Kristen Petra.
5. Segenap dosen dan staf pengajar di Program Studi Teknik Informatika Universitas Kristen Petra.
6. Keluarga yang telah banyak memberikan dukungan doa dan motivasi hingga penulis mampu menyelesaikan tugas akhir guna meraih gelar kesarjanaan ini.
7. Teman-teman tercinta yang telah menempuh perjalan skripsi bersama.
8. Pihak-pihak lain yang telah memberikan bantuan secara langsung maupun tidak langsung dalam pembuatan tugas akhir ini yang tidak dapat disebutkan satu per satu.

Peneliti menyadari bahwa penulisan skripsi ini masih jauh dari sempurna. Oleh karena itu, penulis mengharapkan segala petunjuk, kritik, dan saran yang membangun dari pembaca agar dapat menunjang pengembangan dan perbaikan penulisan.

Akhir kata, penulis mohon maaf apabila ada kekurangan dalam penulisan tugas akhir ini dan penulis dengan senang hati menerima saran dan kritik yang membangun dari pembaca.

Surabaya, Juni 2021

Penulis

# ABSTRAK

Jehezkiel Hardwin Tandijaya:

Skripsi

Implementasi Web Scraping dan Klasifikasi dalam Pembuatan Portal Berita Online

# ABSTRACT

Jehezkiel Hardwin Tandijaya:  
Skripsi

Implementasi Web Scraping dan Klasifikasi dalam Pembuatan Portal Berita Online

# DAFTAR ISI

[NO: 20011538/KOM/2021 i](#_Toc73188693)

[LEMBAR PENGESAHAN ii](#_Toc73188694)

[LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS iii](#_Toc73188695)

[KATA PENGANTAR iv](#_Toc73188696)

[ABSTRAK vi](#_Toc73188697)

[ABSTRACT vii](#_Toc73188698)

[DAFTAR ISI viii](#_Toc73188699)

[DAFTAR GAMBAR x](#_Toc73188700)

[DAFTAR TABEL xi](#_Toc73188701)

[DAFTAR SEGMEN PROGRAM xii](#_Toc73188702)

[1. PENDAHULUAN 1](#_Toc73188703)

[1.1. Latar Belakang 1](#_Toc73188704)

[1.2. Rumusan Masalah 3](#_Toc73188705)

[1.3. Tujuan 3](#_Toc73188706)

[1.4. Ruang Lingkup 3](#_Toc73188707)

[1.5. Metodologi Penelitian 4](#_Toc73188708)

[1.6. Sistematika Penulisan 4](#_Toc73188709)

[2. LANDASAN TEORI 6](#_Toc73188710)

[2.1. Tinjauan Pustaka 6](#_Toc73188711)

[2.1.1. Web Scraping 6](#_Toc73188712)

[2.1.2. Text Mining 7](#_Toc73188713)

[2.1.3. Preprocessing 9](#_Toc73188714)

[2.1.4. Bidirectional Encoder Representations from Transformers 9](#_Toc73188715)

[2.2. Tinjauan Studi 10](#_Toc73188716)

[2.2.1. Contextual Semantic Embeddings based on Fine-tuned AraBERT Model for Arabic Text Multi-class Categorization (Ouatik et al., 2021) 10](#_Toc73188717)

[2.2.2. CyberBERT: BERT for cyberbullying identification (Paul & Saha, 2020) 11](#_Toc73188718)

[2.2.3. Klasifikasi Berita Olahraga Menggunakan Metode *Naïve Bayes* dengan Enhanced Confix Stripping Stemmer (Pramudita et al., 2018) 12](#_Toc73188719)

[3. ANALISIS DAN DESAIN SISTEM 14](#_Toc73188720)

[3.1. Analisis Data 14](#_Toc73188721)

[3.2. Analisis Sistem 15](#_Toc73188722)

[3.2.1. Web Scraping 16](#_Toc73188723)

[3.2.2. Preprocessing 16](#_Toc73188724)

[3.2.3. Klasifikasi 17](#_Toc73188725)

[3.3. Desain Website 19](#_Toc73188726)

[4. IMPLEMENTASI SISTEM 22](#_Toc73188727)

[4.1. Implementasi Web Scraping 22](#_Toc73188728)

[4.2. Implementasi Preprocessing 31](#_Toc73188729)

[4.3. Implementasi Klasifikasi 33](#_Toc73188730)

[4.3.1. Pengolahan Dataset 34](#_Toc73188731)

[4.3.2. Bidirectional Encoder Representations from Transformers 39](#_Toc73188732)

[4.4. Konfigurasi Scheduler 42](#_Toc73188733)

[4.4.1. Prediksi berita baru 43](#_Toc73188734)

[4.5. Aplikasi 45](#_Toc73188735)

[4.5.1. Navbar 45](#_Toc73188736)

[4.5.2. Tampilan Home 46](#_Toc73188737)

[4.5.3. Tampilan Detail Berita 47](#_Toc73188738)

[4.5.4. Tampilan Search 48](#_Toc73188739)

[5. PENGUJIAN SISTEM 50](#_Toc73188740)

[5.1. Pengujian Aplikasi 50](#_Toc73188741)

[5.1.1. Pengujian web scraping 50](#_Toc73188742)

[5.2. Pengujian model 52](#_Toc73188743)

[5.2.1. Pengujian Learning Rate 53](#_Toc73188744)

[5.2.2. Pengujian Dropout 53](#_Toc73188745)

[6. DAFTAR REFERENSI 55](#_Toc73188746)

# DAFTAR GAMBAR

[Gambar 2.1 Teknologi web scraping. 7](#_Toc73188747)

[Gambar 2.2 Arsitektur text mining. 8](#_Toc73188748)

[Gambar 2.3 Arsitektur transformer. 10](#_Toc73188749)

[Gambar 3.1 Contoh data yang akan digunakan. 14](#_Toc73188750)

[Gambar 3.2 Arsitektur sistem 15](#_Toc73188751)

[Gambar 3.3 Tahapan web scraping 16](#_Toc73188752)

[Gambar 3.4 Tahapan preprocessing 17](#_Toc73188753)

[Gambar 3.5 Tahapan klasifikasi BERT 18](#_Toc73188754)

[Gambar 3.6 Tampilan home 19](#_Toc73188755)

[Gambar 3.7 Detail berita 20](#_Toc73188756)

[Gambar 3.8 Hasil search 20](#_Toc73188757)

[Gambar 4.1 Output segmen program 4.4 31](#_Toc73188758)

[Gambar 4.2 Output Segmen Program 4.6 33](#_Toc73188759)

[Gambar 4.3 Attention Mask 37](#_Toc73188760)

[Gambar 4.4 Input Ids dan Kategori 37](#_Toc73188761)

[Gambar 4.5 Membuat task scheduler 42](#_Toc73188762)

[Gambar 4.6 Membuat trigger 43](#_Toc73188763)

[Gambar 4.7 Membuat aksi yang ingin dilakukan 43](#_Toc73188764)

[Gambar 5.1 Pengujian scraping berita 50](#_Toc73188765)

[Gambar 5.2 Halaman detail berita 51](#_Toc73188766)

[Gambar 5.3 Halaman tag yang dipilih 52](#_Toc73188767)

[Gambar 5.4 Halaman search 52](#_Toc73188768)

# DAFTAR TABEL

[Tabel 4‑1 Inisialisasi hyperparameter awal 35](#_Toc73188769)

[Tabel 4‑2 Input Representations BERT 38](#_Toc73188770)

[Tabel 5‑1 Pengujian learning rate 53](#_Toc73188771)

[Tabel 5‑2 Pengujian dropout 53](#_Toc73188772)

# DAFTAR SEGMEN PROGRAM

[Segmen Program 4.1 Import library dan inisialisasi variabel 22](#_Toc72670489)

[Segmen Program 4.2 Fungsi scraping kompas.com 23](#_Toc72670490)

[Segmen Program 4.3 fungsi scraping sindonews.com 24](#_Toc72670491)

[Segmen Program 4.4 Convert to csv 30](#_Toc72670492)

[Segmen Program 4.5 Fungsi preprocessing text 31](#_Toc72670493)

[Segmen Program 4.6 Ubah menjadi list kategori dan Convert ke csv 32](#_Toc72670494)

[Segmen Program 4.7 Cek GPU 33](#_Toc72670495)

[Segmen Program 4.8 Menghubungkan ke Google Drive 34](#_Toc72670496)

[Segmen Program 4.9 Mengubah kategori menjadi list 34](#_Toc72670497)

[Segmen Program 4.10 Inisialisasi hyperparameter 35](#_Toc72670498)

[Segmen Program 4.11 Menentukan rata-rata panjang token 36](#_Toc72670499)

[Segmen Program 4.12 Input Representation BERT 36](#_Toc72670500)

[Segmen Program 4.13 Dataloader 38](#_Toc72670501)

[Segmen Program 4.14 model BERT 39](#_Toc72670502)

[Segmen Program 4.15 Konfigurasi optimizer, scheduler 40](#_Toc72670503)

[Segmen Program 4.16 Fungsi training 40](#_Toc72670504)

[Segmen Program 4.17 Fungsi evaluasi 41](#_Toc72670505)

[Segmen Program 4.18 Menjalankan fungsi training dan evaluasi 42](#_Toc72670506)

[Segmen Program 4.19 Load model yang telah dibuat 45](#_Toc72670507)

[Segmen Program 4.20 Fungsi untuk melakukan prediksi suatu berita 45](#_Toc72670508)

[Segmen Program 4.21 Insert to database 46](#_Toc72670509)

[Segmen Program 4.22 Menghubungkan dengan database 47](#_Toc72670510)

[Segmen Program 4.23 Navbar 47](#_Toc72670511)

[Segmen Program 4.24 SQL untuk menampilkan berita home 48](#_Toc72670512)

[Segmen Program 4.25 Home interface 48](#_Toc72670513)

[Segmen Program 4.26 SQL detail berita 49](#_Toc72670514)

[Segmen Program 4.27 Detail interface 49](#_Toc72670515)

[Segmen Program 4.28 SQL search 50](#_Toc72670516)

[Segmen Program 4.29 Search interface 50](#_Toc72670517)

# PENDAHULUAN

## Latar Belakang

Informasi merupakan salah satu kebutuhan bagi manusia.Berita merupakan salah satu satu sumber informasi mengenai kejadian terkini yang ada pada media massa seperti surat kabar, televisi, dan media lainnya (Kasanah et al., 2019). Seiring berkembangnya teknologi internet menyebabkan kemudahan untuk mendapatkan suatu informasi karena dapat diakses oleh siapa saja. Teknologi internet juga membantu manusia dengan memberikan kemudahan untuk mendapatkan beragam informasi yang diinginkan yang dapat diakses dari berbagai *platform* berita online. *Platform* berita online membantu mengatasi batasan temporal dan spasial dari surat kabar cetak tradisional (Kwak et al., 2020) dan memudahkan kita untuk mencari berbagai informasi dari dalam maupun luar negeri yang diinginkan. Namun, berita yang tersedia sekarang sangat banyak di berbagai platform berita online dengan alamat URL yang berbeda-beda(Aziz & Rahmah, 2017). Belum juga berita yang diakses di salah satu sumber memiliki kredibilitas yang tinggi terhadap suatu kejadian, karena penerbit menggunakan informasi yang salah dan menyesatkan untuk memajukan kepentingan mereka (Aldwairi & Alwahedi, 2018) atau yang dikenal dengan *fake news*. *Fake news* sangat banyak di dunia digital ini, bahkan beberapa pejabat dan individu terlibat dalam penyebarannya agar sesuai dengan tujuan mereka (Apuke & Omar, 2021). Sehingga untuk mengecek kredibilitas dari suatu kejadian, dibutuhkan membaca dari sumber berita lain tidak hanya di satu tempat saja agar tidak terpengaruh dengan berita yang salah. Namun, hal tersebut kurang efektif karena pembaca harus mencari sumber berita lain dengan alamat URL yang berbeda. Oleh karena itu, dibutuhkan suatu *website* yang berisi kumpulan berita dari berbagai sumber berita online.

Berdasarkan beberapa *website* penyedia berita online yang ada di Indonesia, ada beberapa fitur dasar yang membantu mempermudah pembaca mendapatkan berita yang ingin dibaca. Fitur tersebut meliputi menampilkan semua isi berita, menampilkan berita terbaru, mencari berita yang diinginkan. Fitur dasar merupakan fitur yang pasti dimiliki oleh penyedia *website*. Memberikan komentar terhadap berita bukan fitur dasar dikarenakan ada *website* berita yang tidak memiliki fitur tersebut (seperti JawaPos.com, viva.com, dll).

*Bidirectional Encoder Representations from Transformers* merupakan teknik pembelajaran mesin yang dikembangkan oleh Google berbasis Transformer untuk pra-pelatihan pemrosesan Bahasa alami (Natural Language Processing). Adapun penelitian – penelitian sebelumnya *Contextual Semantic Embeddings based on Fine-tuned AraBERT Model for Arabic Text Multi-class Categorization* dimana membandingkan AraBERT *pre-trained model* dan melakukan *fine-tunes* dengan AraBERT sebagai *feature extractor* model dalam hubungannya dengan beberapa neural network classifier dan SVM untuk melakukan klasifikasi (Ouatik et al., 2021). *CyberBERT: BERT for cyberbullying identification* dimana model BERT menghasilkan peforma lebih baik dari model klasifikasi baik *machine learning* maupun *deep learning* dalam melakukan klasifikasi *cyberbullying* (Paul & Saha, 2020). *News Classification Using Naïve Bayes and Two-Phase Feature Selection Model* dimana data didapat di kompas.com dan peneliti melakukan klasifikasi dengan naïve bayes dan 2 fase feature selection dimana akurasi tertinggi didapat sebesar 86% (Ali Fauzi et al., 2017) .*Efficient classification model of web news documents using machine learning algorithms for accurate information* dimana penulis membandingkan *K-Nearest Neighbor*(kNN), *Support Vector Machine* (SVM), *Decision Tree* (DT), dan *Long Short-Term Memory* (LSTM) dan SVM menghasilkan akurasi paling tinggi sebesar 95.04% dan akurasi paling rendah didapat KNN sebesar 88.72% (Mulahuwaish et al., 2020). Klasifikasi Berita Olahraga Menggunakan Metode *Naïve Bayes* Dengan *Enhanced Confix Stripping Stemmer* dimana berita dapat diklasifikasi dengan menggunakan metode *naïve bayes* dengan *confix stripping stemmer* menghasilkan akurasi sebesar 77% (Pramudita et al., 2018). Adapun kelebihan dan kekurangan metode-metode yang digunakan sebelumnya. *K-Nearest Neighbor* (KNN) mahal secara komputasi saat diterapkan dalam data berdimensi tinggi terutama jika set pelatihannya banyak, rentan terhadap overfit dan kinerja relatif rendah saat dihadapkan oleh teks panjang dan memiliki banyak fitur (Hartmann et al., 2019). Namun, kNN lebih sering digunakan daripada metode lain untuk menangani teks yang pendek (Hartmann et al., 2019) . *Naïve Bayes* (NB) bekerja dengan baik ketika fitur saling bergantung, model yang dihasilkan mudah untuk direpresentasikan dan dijelaskan, dan direkomendasikan untuk ukuran sampel yang lebih kecil karena regularisasi yang melekat, membuatnya cenderung tidak terlalu pas dibandingkan dengan pengklasifikasi diskriminatif (Hartmann et al., 2019). *Support Vector Machine* (SVM) dapat menangani dengan baik data berdimensi tinggi, tidak rentan terhadap *overfitting,* tetapi representasi SVM yang terbatas dapat mengakibatkan kurangnya kemampuan untuk memodelkan pola bernuansa dalam data pelatihan (Hartmann et al., 2019). Oleh karena itu, pada skripsi metode deep learning yang digunakan adalah Bidirectional Encoder Representations from Transformers karena metode klasifikasi dengan menggunakan pendekatan tradisional machine learning memiliki batasan tertentu dalam pelatihan dataset skala besar walaupun memiliki karakteristik efisiensi dan stabilitas tinggi (Sari et al., 2017).

Untuk menjawab permasalahan diatas, dibutuhkan *website* yang berisi kumpulan berita dari berbagai sumber berita online dengan fitur dasar yang dapat memudahkan pembaca untuk membaca dari satu *website* dengan berita-berita yang disediakan oleh sumber lain.

## Rumusan Masalah

* Apakah tingkat akurasi dalam mengklasifikasi berita dapat mencapai 80% untuk mengklasifikasi text bahasa indonesia ?
* Bagaimana tingkat akurasi yang dihasilkan oleh BERT dalam mengklasifikasi text Bahasa Indonesia

## Tujuan

Tujuan dari penulisan skripsi ini adalah membuat *website* yang berisi kumpulan berita dari berbagai sumber berita online dengan fitur dasar yang dapat memudahkan pembaca untuk membaca dari satu *website* dengan berita-berita yang disediakan oleh sumber lain.

## Ruang Lingkup

Ruang lingkup dibatasi pada:

* Aplikasi dalam bentuk *website*.
* Fitur meliputi fitur basic yaitu menampilkan semua isi berita, menampilkan berita terbaru, mencari berita yang diinginkan.
* Dataset berupa text berformati .csv berupa berita yang didapat dengan *Web Scraping d*engan kurun waktu 1 bulan berisi judul berita, url, isi berita beserta kategori berita berupa integer yang sudah diinisialisasi*.* perbandingan data training dan testing (untuk evaluasi model) adalah 80:20 dengan library scikit-learn.
* Situs yang digunakan untuk konten dalam portal berita adalah situs berita berbahasa indonesia.
* Situs berita yang digunakan sindonews.com dan kompas.com.
* Kategori yang dianalisa adalah *health*, edukasi, teknologi, *lifestyle* dan *sport*.
* Menggunakan *library BeautifulSoup* untuk melakukan *web scraping .*
* Menggunakan MySQL untuk menyimpan kumpulan berita yang akan digunakan untuk menjadi konten beserta hasil klasifikasi.
* Output merupakan hasil klasifikasi dari konten berita yang dimasukkan (bisa lebih dari 1 kategori)
* Metrik Pengujian : akurasi, precision, recall, f1score. (confusion matrix)
* Arsitektur Sistem

Input

Text Preprocessing

Web Scraping

Klasifikasi

* Text Preprocessing meliputi :
  + *Filter string* : melakukan pembersihan pada *string* seperti menghapus tanda baca, angka, *special character* dan sebagainya.
  + *Stopword Removal :* proses dimana kata – kata yang umum dan sering muncul dihilangkan seperti kata penghubung, dll) atau kata-kata yang tidak penting yang didefinisikan oleh pembuat program.
  + *Stemming* : mengubah semua kata yang memiliki imbuhan menjadi kata dasar.
* Metode yang digunakan adalah *Bidirectional Encoder Representations from Transformers*

## Metodologi Penelitian

Dalam melakukan penelitian, terdapat beberapa langkah yang harus dilakukan, yaitu:

1. Studi literatur mengenai
   1. *Web Scraping*
   2. *Text Mining*
   3. *Preprocessing*
   4. Metode *Bidirectional Encoder Representations from Transformers*
2. Pengumpulan dataset
   1. Pengumpulan dataset berupa berita yang didapat dari hasil *web scraping*.
3. Perencanaan dan Pembuatan Program:
   1. Melakukan *Web Scraping*
   2. Melakukan *preprocessing*
   3. Implementasi *Bidirectional Encoder Representations from Transformers*
4. Pengujian dan Analisa :
   1. Melakukan test terhadap program yang telah dibuat
   2. Analisa *output* pada program
5. Pengambilan kesimpulan dan Pembuatan laporan.

## Sistematika Penulisan

Adapun sistematika penulisan yang digunakan untuk Menyusun skripsi ini adalah sebagai berikut:

BAB I : PENDAHULUAN

Bab ini membahas latar belakang permasalahan, perumusan masalah, tujuan skripsi, ruang lingkup, metode penelitian yang digunakan, dan relevansi skripsi ini.

BAB II : LANDASAN TEORI

Bab ini membahas mengenai teori-teori yang menjadi landasan dalam kegiatan penelitian ini, seperti teori *web scraping, preprocessing, text mining,* metode *Bidirectional Encoder Representations from Transformers (BERT)*.

BAB III : ANALISIS DAN DESAIN SISTEM

Bab ini membahas analisis data yang akan dipakai, *flowchart* alur kerja metode, dan desain *website*.

BAB IV : IMPLEMENTASI SISTEM

Bab ini berisikan kebutuhan implementasi sistem, konfigurasi *scheduler, user* *interface* dan membahas secara detail hasil pembuatan sistem di setiap halaman sistem.

BAB V : PENGUJIAN DAN EVALUASI SISTEM

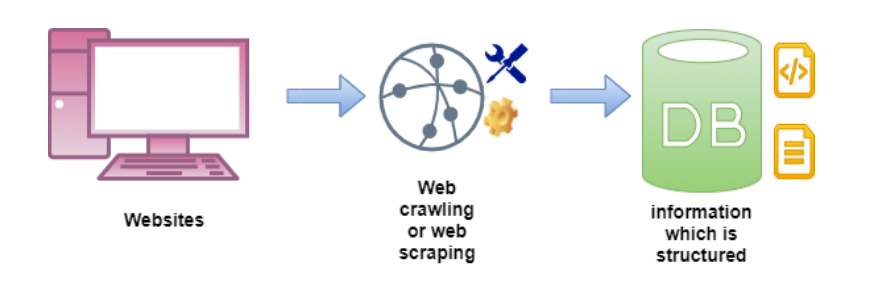
# LANDASAN TEORI

Bab ini akan memberikan penjelasan yang berkaitan dengan teori apa saja yang digunakan dalam pembuatan portal berita. Berikut ini merupakan penjelasan tentang teori-teori tersebut.

## Tinjauan Pustaka

### Web Scraping

*Web Scraping* atau yang dikenal sebagai ekstraksi *web* adalah teknik untuk mengekstraksi data dari *World Wide Web* (WWW) dan menyimpan ke dalam database, kemudian data tersebut nantinya dapat diambil kembali untuk dilakukan analisa (Flores et al., 2020). Tujuan dilakukannya *web scraping* agar orang yang menggunakan *web scraping* dapat lebih mudah untuk mendapatkan suatu data yang dibutuhkan dalam jumlah yang besar, tidak perlu melakukan teknik *copy* dan *paste* karena teknik tersebut akan membutuhkan waktu yang lama (apabila digunakan untuk mengambil data yang cukup besar). Belakangan ini, banyak peneliti yang menggunakan teknik *web scraping* karena memiliki banyak manfaat, seperti untuk membuat kumpulan data mereka sendiri untuk ekstraksi informasi artikel jurnal, proyek terkait *text mining*, dll (Karthikeyan et al., 2019). Data yang didapatkan beragam sesuai dengan apa yang diinginkan. Sebagai contoh untuk melakukan klasifikasi berita, dengan menggunakan teknik *web scraping* akan memudahkan kita untuk mendapatkan data (seperti judul berita, isi berita beserta kategori dalam bentuk .csv) dengan jumlah yang diingkan yang akan diolah untuk membuat suatu knowledge baru. Ada 2 tipe web scraping. Pertama *screen scraping* dimana kita dapat mengekstrak data dari kode sebuah *website* dengan *HTML Parser* atau *regular expression (regex)* (Dogucu & Çetinkaya-Rundel, 2020). Kedua dengan menggunakan antarmuka pemrograman aplikasi atau yang dikenal dengan API dimana sebuah *website* menawarkan sekumpulan *HTTP Request* yang membalikkan nilai JSON atau XML (Dogucu & Çetinkaya-Rundel, 2020). Manfaat dan tujuan web scraping .*Web Scraping* berbeda dengan *Web Crawler,* dimana *Web Crawler* mengunjungi semua situs utama yang akan dijadikan patokan untuk melakukan *Web Crawler*, sedangkan *Web Scraping* hanya mengambil informasi dari situs yang ingin dituju sesuai dengan keinginan pengguna (Setiawan et al., 2020). Berikut menunjukkan gambaran umum bagaimana web scraping dilakukan.



Gambar 2.1 Teknologi web scraping.

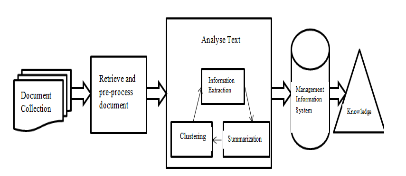
Sumber : BABU, K. (2020). Survey on Web scraping technology. Waffen-Und Kostumkunde Journal, 16(06), 1–8.

Ada beberapa tahapan yang dilakukan untuk mengambil data secara otomatis dari sebuah *website* adalah sebagai berikut (Priyanto & Ma’arif, 2018):

1. Mempelajari dokumen HTML dari *website* yang akan diambil informasinya
2. Menelusuri mekanisme kerja dari navigasi pada suatu *website* yang akan diambil informasinya untuk ditirukan pada aplikasi *web scrapper* yang akan dibuat
3. Berdasarkan 2 langkah sebelumnya, aplikasi *web scrapper* dibuat untuk mengotomatisasi pengambilan informasi dari *website* yang ditentukan
4. Informasi yang didapat akan disimpan dalam sebuah *database*.

### Text Mining

Text Mining adalah proses penggalian informasi dari suatu text baik text yang *structured text*, *semi-structured* (JSON & XML), dan *unstructured text (documents, videos, images)* (Allahyari et al., 2017). *Text Mining* digunakan untuk mengekstrak fakta dan hubungan dalam bentuk terstruktur yang digunakan untuk membuat anotasi database khusus yang bertujuan untuk mengekstrak pengetahuan antara domain dan lebih umum dalam intelijen bisnis untuk mendukung pengambilan keputusan strategis (Westergaard et al., 2018). *Text Mining* dan *Data Mining* memiliki perbedaan dimana *Data Mining* metode dibuat untuk menangani data terstruktur yang berasal dari database (Al Sbou, 2018). Gambar dibawah menunjukkan bagaimana text mining bekerja untuk mendapatkan suatu knowledge baru.



Gambar 2.2 Arsitektur text mining.

Sumber : Tandel, S. S., Jamadar, A., & Dudugu, S. (2019). A Survey on Text Mining Techniques. 2019 5th International Conference on Advanced Computing and Communication Systems, ICACCS 2019, Icaccs, 1022–1026.

Gambar 2.2 menunjukkan bagaimana cara text mining mengubah *unstructured data* menjadi *structured data* yang diambil dari beberapa sumber seperti *plain text, web pages,* dan *data files.* Kemudian dilakukan *preprocessing* dan *cleansing* yang diimplementasikan untuk mendeteksi dan menghilangkan anomali. *Cleansing* berguna untuk menghilangkan imbuhan (mengubah ke kata dasar / *word root*) dan mengenali pola dari data terstruktur. Setelah dilakukan *preprocessing dan cleasing*, akan dilakukan analisa teks dengam menggunakan teknik *text mining* untuk menghasilkan sebuah knowledge baru.

Text mining bisa dimanfaatkan untuk melakukan *sentiment analysis, text classification, detection email spam,* dll. Klasifikasi teks adalah proses menentukan satu atau lebih kelas atau kategori yang telah ditentukan sebelumnya kedalam dokumen yang akan dianalisis (Boukil et al., 2018). Ada 3 tahapan dasar dari tugas klasifikasi teks : *data pre-processing*, *text classification* dan *evaluation* (Sayed et al., 2019)*. Text Classification* dapat dilakukan dengan *machine learning* maupun *deep learning*. Dalam tahapan *text classification*, terdapat tahapan *feature extraction* dimana bertujuan untuk membuat *vector space* berdasarkan konteksnya. Setelah melakukan klasifikasi, perlu dilakukan evaluasi yang bertujuan untuk mengukur model yang telah dibuat bekerja. Langkah-langkah evaluasi multi-label yang ada pada *confusion matrix* dalam persamaan berikut (Sari et al., 2020):

|  |  |
| --- | --- |
|  | (2.1) |

|  |  |
| --- | --- |
|  | (2.2) |

|  |  |
| --- | --- |
|  | (2.3) |

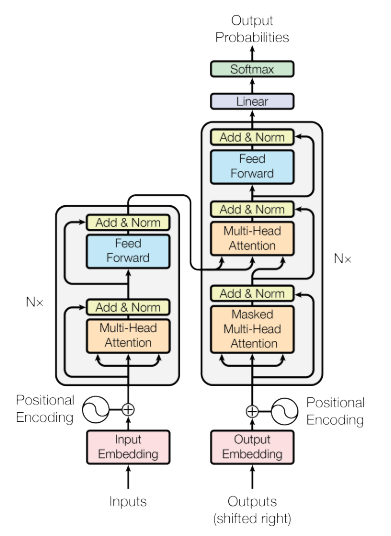
|  |  |
| --- | --- |
|  | (2.4) |

### Preprocessing

*Preprocessing* memiliki peranan yang sangat penting dalam aplikasi *text mining* (Kannan et al., 2015). *Preprocessing* dapat membersihkan data dari *noise* seperti kesalahan ejaan, pengurangan karakter yang direplikasi dan disambiguasi akronim yang ambigu) (HaCohen-Kerner et al., 2020). Pada umumnya, terdapat beberapa proses *preprocessing* seperti *case folding, tokenization, stopwords removal dan stemming* (Sistem, 2021) *. Case Folding* mengubah semua text menjadi huruf kecil. *Tokenization* merupakan proses dimana teks dipecah menjadi sebuah kata, frasa, simbol atau elemen lainnya yang disebut *token*. *Stopword Removal* merupakan proses dimana kata – kata yang umum dan sering muncul dihilangkan seperti kata penghubung, dll) atau kata-kata yang tidak penting yang didefinisikan oleh pembuat program. *Stemming* adalah salah satu fase dalam pra-pemrosesan teks yang menerapkan bahasa alami untuk menghilangkan imbuhan dari kata untuk mengubahnya menjadi kata dasar (Utomo et al., 2020). Contoh kata “mengunjungi” menjadi kata “kunjung”. *Filtering* adalah melakukan pembersihan pada *string* seperti menghapus tanda baca, angka, *special character* dan sebagainya.

### Bidirectional Encoder Representations from Transformers

*Bidirectional Encoder Representations from Transformers* merupakan model representasi kata kontekstual yang dilatih sebelumnya berdasarkan MLM (Masked Language Model), menggunakan *Transformers* dua arah (Peng et al., 2019). Arsitektur model BERT adalah struktur encoder-decorder transformator dua arah multi-layer (Fang et al., 2020). Transformer mengikuti keseluruhan arsitektur ini menggunakan *stacked self-attention* dan *point-wise*, terhubung sepenuhnya untuk encoder dan decoder (Vaswani et al., 2017). Masing-masing ditunjukkan pada Gambar 2.3:



Gambar 2.3 Arsitektur transformer.

Sumber : Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 2017-Decem(Nips), 5999–6009.

Transformer memiliki 2 komponen yaitu *encoder* dan *decoder*. *Encoder* mengambil kata dan menghasilkan *embedding* dari kata tersebut. E*mbedding* ini berupa vektor yang berguna untuk merangkum apa arti dari kata tersebut. D*ecoder* mengambil *embedding* dari *encoder,* kemudian mengeluarkannya sebagai suatu kata pada satu waktu sampai membuat suatu kalimat. Sebagai contoh, ingin menerjemahkan dari inggris ke Indonesia. *Encoder* akan mengambil kata “*This is amazing*” kemudian akan diubah menjadi embedding berupa vektor dari kata “*This is amazing*” kemudian decoder akan mengambil embedding dari pembuat encoder sebelumnnya dan kata yang dihasilkan dari terjemahan tersebut dan menggunakannya untuk menghasilkan kata terjemahan selanjutnya. Terjemahan tersebut akan di *generate* terus menerus hingga mendapatkan kalimat yang sebenarnya. Dalam hal ini, *encoder* belajar konteks dari Bahasa tersebut sedangkan *decoder* belajar bagaimana kata-kata inggris berhubungan dengan kata-kata Indonesia.

Ada 2 langkah dalam kinerja framework BERT yaitu *pre-training* dan *fine-tuning.* Pre-training BERT tidak menggunakan cara tradisional left-to-right atau right-to-left, tetapi menggunakan Masked Language Modeling (MLM) dan Next Sentence Prediction (NSP) untuk pre-training data (Devlin et al., 2019). MLM mengisi tempat kosong, dimana model menggunakan context word disekitar mask token untuk memprediksi kata yang seharusnya sedangkan NSP merupakan prediksi kalimat berikutnya dengan dua model yang diberikan. Setelah melakukan pre-training data, BERT akan melakukan fine-tuning dimana fine-tuning diinisialisasi dengan parameter yang telah dilatih sebelumnya, dan semua parameter fine-tuning menggunakan data berlabel dari tugas-tugas downstream.

## Tinjauan Studi

### Contextual Semantic Embeddings based on Fine-tuned AraBERT Model for Arabic Text Multi-class Categorization (Ouatik et al., 2021)

* 1. Masalah :

Dengan terobosan baru-baru ini dalam Natural Language Processing (NLP), Kategorisasi Teks Arab (ATC) telah menjadi bidang penelitian yang aktif karena bahasa Arab menghadirkan berbagai tantangan dalam hal morfologi dan strukturnya yang kompleks.Pemodelan teks tradisional berdasarkan representasi Bag-Of-Words (BOW) mencapai pertunjukan mutakhir di wilayah NLP. Namun, ia menderita kutukan dimensionalitas dan kurangnya hubungan semantik antara unit teks yang berbeda. Model penyematan kata yang telah dilatih sebelumnya seperti Word2vec, Doc2vec, dan Glove telah muncul sebagai satu set model representasi yang efektif untuk klasifikasi teks dan tugas NLP lainnya. Meskipun penyematan kata telah terbukti performa yang baik, ini tidak mempertimbangkan hubungan antara beberapa kata dan makna kalimat lengkap dalam teks.

* 1. Metode :

Pada penelitian ini, peneliti melakukan text *preprocessing* yang terdiri dari *Remove Punctuation, Nomalizing, Data transformation* (mengubah data agar sesuai dengan *fine-tuned AraBERT model* dimana terdiri dari *tokenization, convert each text to BERT’s format (*[CLS]di awal text dan [SEP] diakhir teks*),* dan mengubah setiap token menjadi angka sesuai dengan indeks yang ada di dalam AraBERT *vocabulary*)*.* Setelah *preprocessing*, peneliti melakukan klasifikasi dengan 2 variasi, *AraBERT Fine-tuning* dan *AraBERT for Feature Extraction*.

* 1. Hasil :

Teks dapat diklasifikasi dan mendapatkan akurasi dengan *proposed method fine-tuned AraBERTv0.1* dengan hasil 99% dan *F1-score* 99% dan akurasi yang didapatkan *proposed fine-tuned AraBERT v1* dengan hasil 99.2% dan *F1-score* 99%. 2 *fine-tuned AraBERT* tersebut terbukti mengalahkan semua *baseline* yang dicoba oleh peneliti tersebut.

### CyberBERT: BERT for cyberbullying identification (Paul & Saha, 2020)

1. Masalah :

Platform media sosial online memungkinkan orang untuk berbagi dan mengekspresikan pikiran dan perasaan mereka secara bebas dan terbuka dengan orang lain. Informasi yang tersedia melalui media sosial ini adalah sumber yang kaya untuk dianalisa sentimen atau menyimpulkan penggunaan dan penyalahgunaan lainnya yang meningkat. Pertumbuhan jejaring sosial yang meningkat ini menyebabkan pelecehan dan penguntitan terus menerus yang biasa disebut sebagai cyber-bullying. Terkadang, tindakan penindasan maya ini bersifat anonim dan memiliki efek yang menghancurkan. Oleh karena itu, mendeteksi cyberbullying pada tahap awal merupakan langkah penting untuk mencegah tindakan ini dan juga untuk menghindari kejadian fatal yang diakibatkannya.

1. Metode :

Pada penelitian ini, peneliti menggunakan dataset twitter yang memiliki 3 kelas yaitu *racism, sexism,* dan *none*, dataset wikipedia memiliki kelas *toxic* atau *non-toxic* berdasarkan konten dan dataset formspiring memiliki kelas *bully* dan *non-bully*. Peneliti mengubah input dan menyesuaikan dengan input yang dibutuhkan BERT (*token embeddings, segment embeddings,* dan *position embedings*)

1. Hasil :

F1-score tertinggi yang dihasilkan oleh metode BERT dalam berbagai dataset adalah 0.94 untuk dataset twitter, 0.91 untuk dataset wikipedia dan 0.93 untuk dataset formspring.

### Klasifikasi Berita Olahraga Menggunakan Metode *Naïve Bayes* dengan Enhanced Confix Stripping Stemmer (Pramudita et al., 2018)

1. Masalah :

Berita adalah informasi berdasarkan fakta atau laporan mengenai suatu kejadian yang sedang atau telah terjadi dan dipublikasikan melalui media cetak, siaran, internet maupun dari mulut ke mulut. Dengan adanya berita, masyarakat menjadi lebih tahu mengenai salah satu berita yang paling banyak diakses oleh masyarakat. Berita Olahraga merupakan salah satu berita yang paling banyak diakses oleh masyarakat. Berbeda dengan media cetak maupun siaran, berita melalui internet dapat diakses kapan saja tanpa adanya batas waktu. Akan tetapi seiring dengan berjalannya waktu, jumlah berita olahraga pada internet akan semakin besar. Besarnya jumlah berita membuat berita tersebut perlu diorganisasi kedalam kelompok atau kategorisasi sesuai dengan isi dari berita. Dengan tujuan agar berita tersebut terorganisir sehingga memudahkan dalam hal akses.

1. Metode :

Pada penelitian ini, peneliti pertama-tama melakukan *preprocessing* terhadap data berita olahraga. Preprocessing terdiri dari *Case folding, Tokenizing, Filtering* dan *Stemming*. Setelah dilakukan *preprocessing* peneliti menghitung *Term Frequency* dan melakukan perbandingan *Term*. Setelah itu peneliti melakukan klasifikasi dengan *Naïve Bayes Classifier*.

1. Hasil :

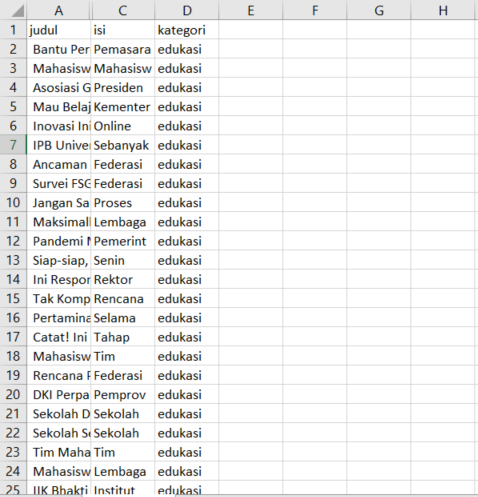
Pengujian dilakukan dengan menggunakan dokumen uji berita olahraga yang diambil dari sport.detik.com dengan jumlah 30 dokumen berita. Berita tersebut disimpan dengan file berformat .txt. Data diambil 18 untuk pengujian dan didapatkan hasil 14 berita kategorinya relevan. Hasil akurasi yang didapat sebesar 77%, dengan *error rate* sebesar 23%.

# ANALISIS DAN DESAIN SISTEM

Bab ini memberikan penjelasan tentang hal – hal yang berkaitan dengan perancangan dan desain sistem. Kedua hal tersebut menjadi gambaran besar tentang cara kerja dari implementasi web scraping dan klasifikasi dalam pembuatan portal berita online. Selain itu, akan dijelaskan tentang data yang akan digunakan, tahapan-tahapan pre-processing, klasifikasi dengan BERT, hingga desain interface dari sistem yang akan dibuat.

## Analisis Data

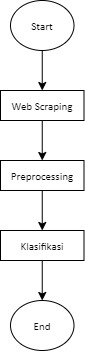
Pada penelitian ini, data yang akan diambil yaitu data dari portal berita berbahasa Indonesia yaitu kompas.com dan sindonews.com dengan menggunakan *web* scraping dengan bantuan library BeautifulSoup. Data yang diambil adalah judul dari berita, isi dari berita dan kategori berita dan data tersebut berformat .csv. Dataset tersebut akan dibagi menjadi data training dan data testing dengan perbandingan 80:20. Contoh data yang akan digunakan dapat dilihat pada Gambar 3.1.



Gambar 3.1 Contoh data yang akan digunakan.

## Analisis Sistem

Pada analisis sistem akan dibahas mengenai proses dari input data yang berupa format .csv yang akan digunakan untuk membuat knowledge baru (klasifikasi). Sistem mencakup pengambilan data dengan bantuan *Web Scraping*, mengolah data yang didapat dengan melalui tahapan *preprocessing* dan tahapan terakhir yang dilakukan adalah klasifikasi dengan menggunakan metode *Bidirectional Encoder Representations from Transformers* (BERT). Arsitektur Sistem dapat dilihat pada Gambar 3.2.

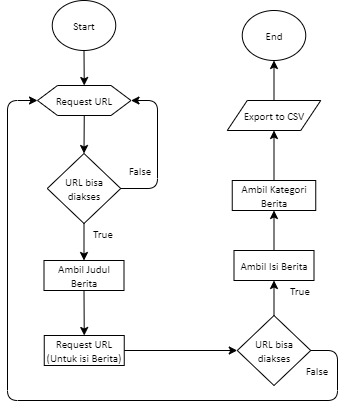


Gambar 3.2 Arsitektur sistem

Arsitektur sistem secara umum dapat dilihat diatas, hal pertama yang dilakukan dalam penelitian ini adalah dengan mengambil data dari portal berita Bahasa Indonesia yaitu kompas.com dan sindonews.com dengan cara *web scraping* dengan bantuan library bs4. Setelah mendapatkan data yang dibutuhkan, dilakukan *preprocessing* yang bertujuan untuk membersihkan data dari *noise*. *Preprocessing* meliputi *filter string* (menghilangkan simbol, angka, hanya menerima string), *stopword removal,* dan *stemming* (mengubah menjadi kata dasar seperti memakan -> makan). Setelah dilakukan *preprocessing*, tahapan terakhir yang dilakukan adalah dengan melakukan klasifikasi dengan menggunakan metode *Bidirectional Encoder Represetantions from Transformers.*

### Web Scraping

Pada penelitian ini, data yang akan diambil yaitu data dari portal berita berbahasa Indonesia yaitu kompas.com dan sindonews.com dengan menggunakan *web* scraping dengan bantuan library BeautifulSoup. Tahapan Web Scraping dapat dilihat pada Gambar 3.3.

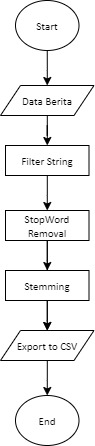


Gambar 3.3 Tahapan web scraping

Pada proses *web scraping*, pertama-tama akan dilakukan *request* terhadap *website* yang akan digunakan. *Request* ini bertujuan untuk mengetahui apakah *website* tersebut diperbolehkan untuk dilakukan *scraping* atau tidak. Setelah mendapatkan respon code [200] dari *website* tersebut, langkah berikutnya adalah mengambil judul berita dari *website* tersebut. Setelah judul berita didapatkan, akan dilakukan *request url* kembali untuk mendapatkan isi berita karena isi berita berada pada url lain. Setelah semua data yang dibutuhkan didapatkan, langkah terakhir yang akan dilakukan adalah dengan mengexport ke format .csv

### Preprocessing

Setelah isi berita didapatkan, akan dilakukan tahapan preprocessing terhadap isi berita tersebut untuk membersihkan data dari *noise*. tahapan *preprocessing* meliputi *filter string* (menghilangkan simbol, angka, hanya menerima string), *Case Folding* (mengubah semua text menjadi huruf kecil) *stopword removal, stemming* (mengubah menjadi kata dasar seperti memakan -> makan).Tahapan preprocessing dapat dilihat pada Gambar 3.4.

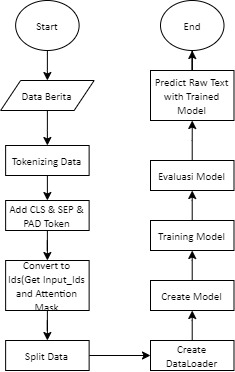


Gambar 3.4 Tahapan preprocessing

Pada proses *preprocessing*, pertama-tama akan dilakukan *filter string* untuk melakukan pembersihan pada *string* seperti menghapus tanda baca, angka, *special character* dan sebagainya. Tahapan ini juga mengubah semua huruf besar menjadi huruf kecil (*Lowercase text*). Setelah itu, akan dilakukan *stopword removal* yang bertujuan untuk menghapus kata-kata yang sering bermunculan dan tidak ada manfaatnya. Langkah terakhir, akan dilakukan *stemming* untuk menghilangkan imbuhan dan merubahnya menjadi kata dasar seperti kata “berkunjung” menjadi “kunjung”. Setelah dilakukan *preprocessing*, data tersebut akan diubah kategorinya menjadi list agar memudahkan pada saat *training* dan diekspor menjadi file dengan ekstensi .csv dikarenakan saat dilakukan *preprocessing* waktu yang dibutuhkan sangat lama sehingga diubah ke csv agar memudahkan untuk melanjutkan ke tahapan berikutnya.

### Klasifikasi

Tahapan terakhir yang akan dilakukan adalah klasifikasi dengan metode *Bidirectional Encoder Representations from Transformers.* Sebelum dilakukan klasifikasi, data perlu diubah ke input yang dapat dibaca oleh metode BERT. Setelah mendapatkan input yang dapat dibaca oleh BERT, akan dilakukan pembuatan model yang sesuai dan akan dilakukan evaluasi terhadap model yang telah dibuat dan dilatih sebelumnya. Secara garis besar tahapan klasifikasi dalam penelitian ini dapat dilihat pada Gambar 3.5.

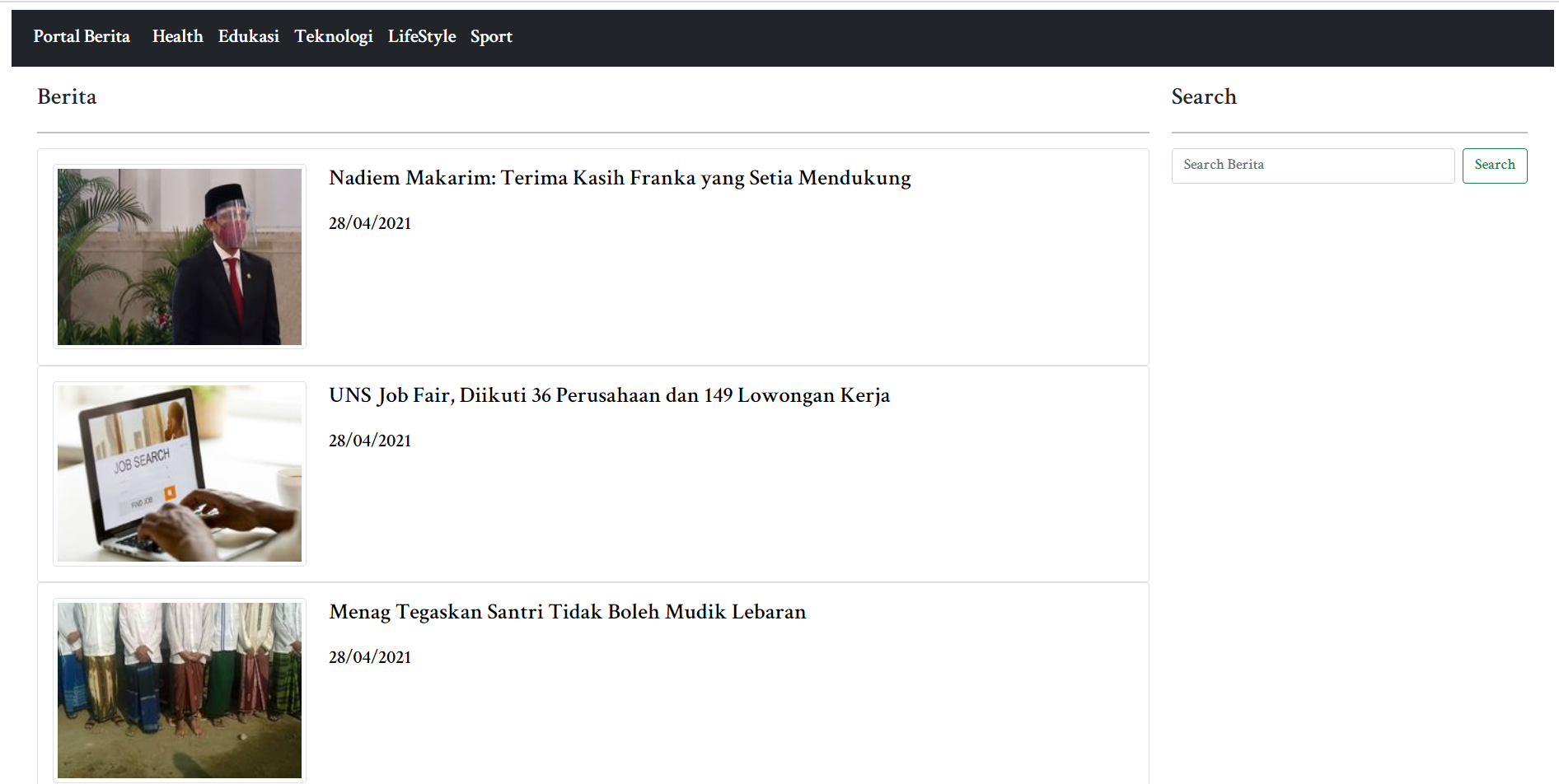


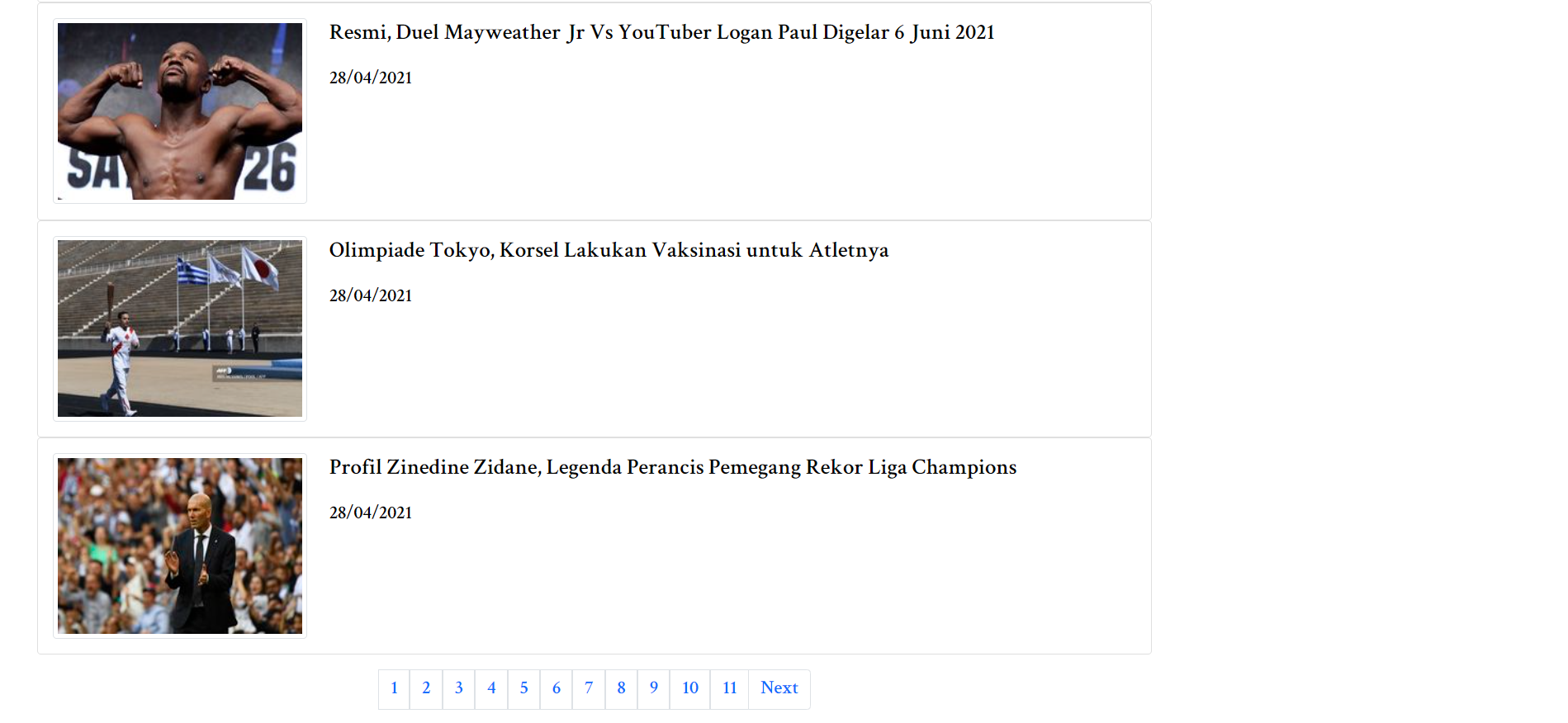
Gambar 3.5 Tahapan klasifikasi BERT

Pada Tahapan klasifikasi, pertama input data yang didapat diubah kedalam bentuk input yang dapat dibaca oleh BERT. Untuk dapat dibaca oleh BERT, perlu ditambahkan token [CLS] di awal kalimat dan token [SEP] diakhir kalimat, dan menentukan panjang kalimat untuk menambahkan [PAD] token sebanyak sisa token. Contoh panjang kalimat yang telah ditentukan berukuran 10 dan contoh kalimat “saya makan nasi” memiliki 5 token (dengan penambahan token [CLS] diawal kalimat dan token [SEP] diakhir kalimat) sehingga token [PAD] yang akan ditambahkan sebanyak 5 di setelah token [SEP]. Setelah token [CLS], [SEP], dan [PAD] ditambahkan, BERT akan mengubah setiap token kata menjadi token ids dan mengembalikan hasil input\_ids dan attention\_mask untuk nantinya akan diteruskan kedalam model BERT. Attention\_maks hanya bernilai 0 dan 1 dimana 1 merupakan token asli dan 0 merupakan token [PAD] yang ditambahkan. Setelah mengubah data menjadi input yang dapat dibaca oleh BERT, akan dilakukan pembuatan *dataloader* yang berguna untuk merandom data pada setiap EPOCH untuk mencegah model dari *overfit* dan membantu mempercepat pengambilan data. Setelah membuat *dataloader*, tahapan selanjutnya adalah pembuatan model. Setelah model dibuat, akan dilakukan training dan evaluasi terhadap model yang telah dibuat. Langkah terakhir yang dilakukan adalah memprediksi text dengan model yang telah di *training* dan menambahkan layer *softmax* yang bertujuan untuk mendapatkan probabilitas dari setiap kelas.

## Desain Website

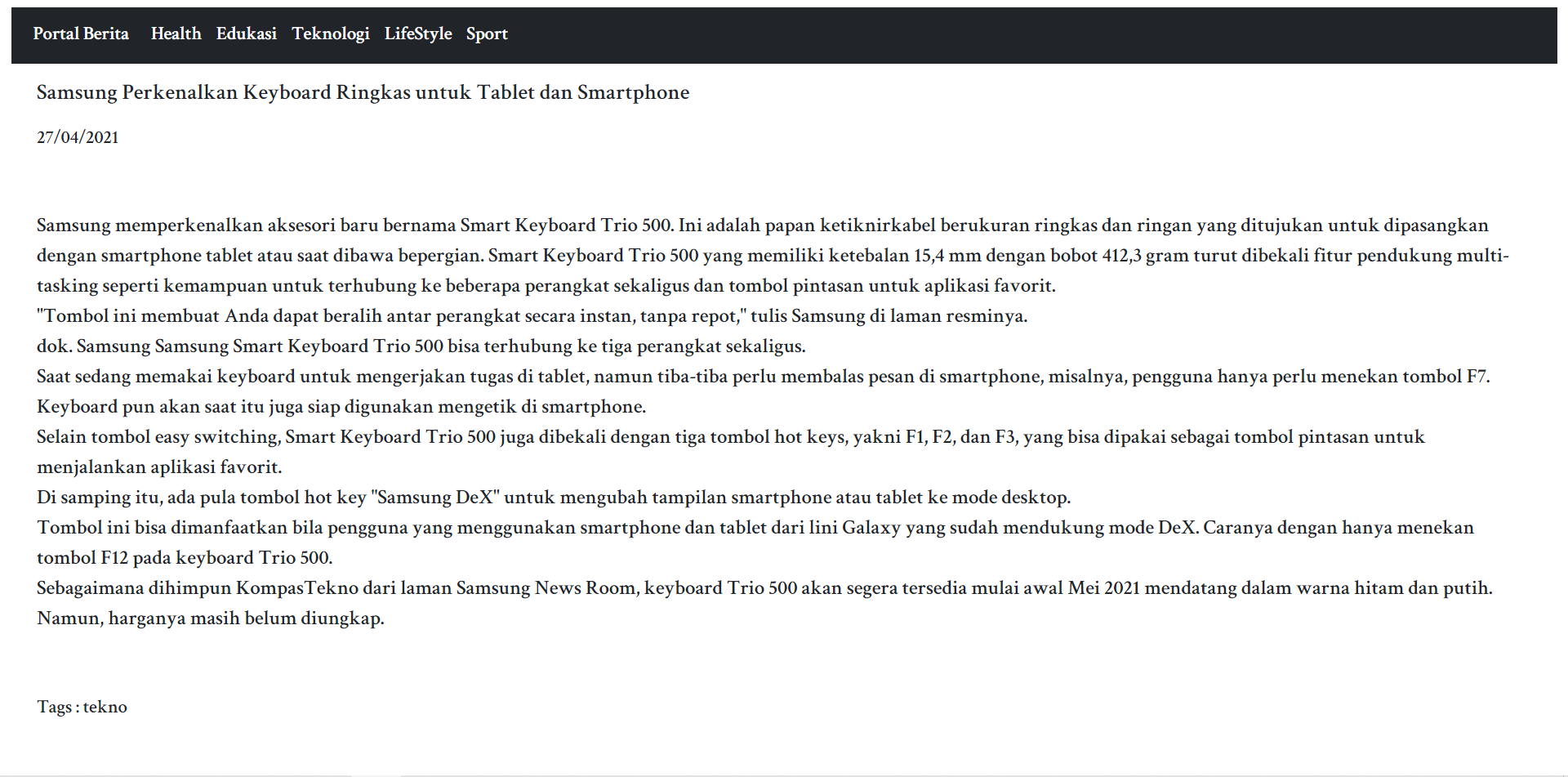
Tampilan *website* akan berupa





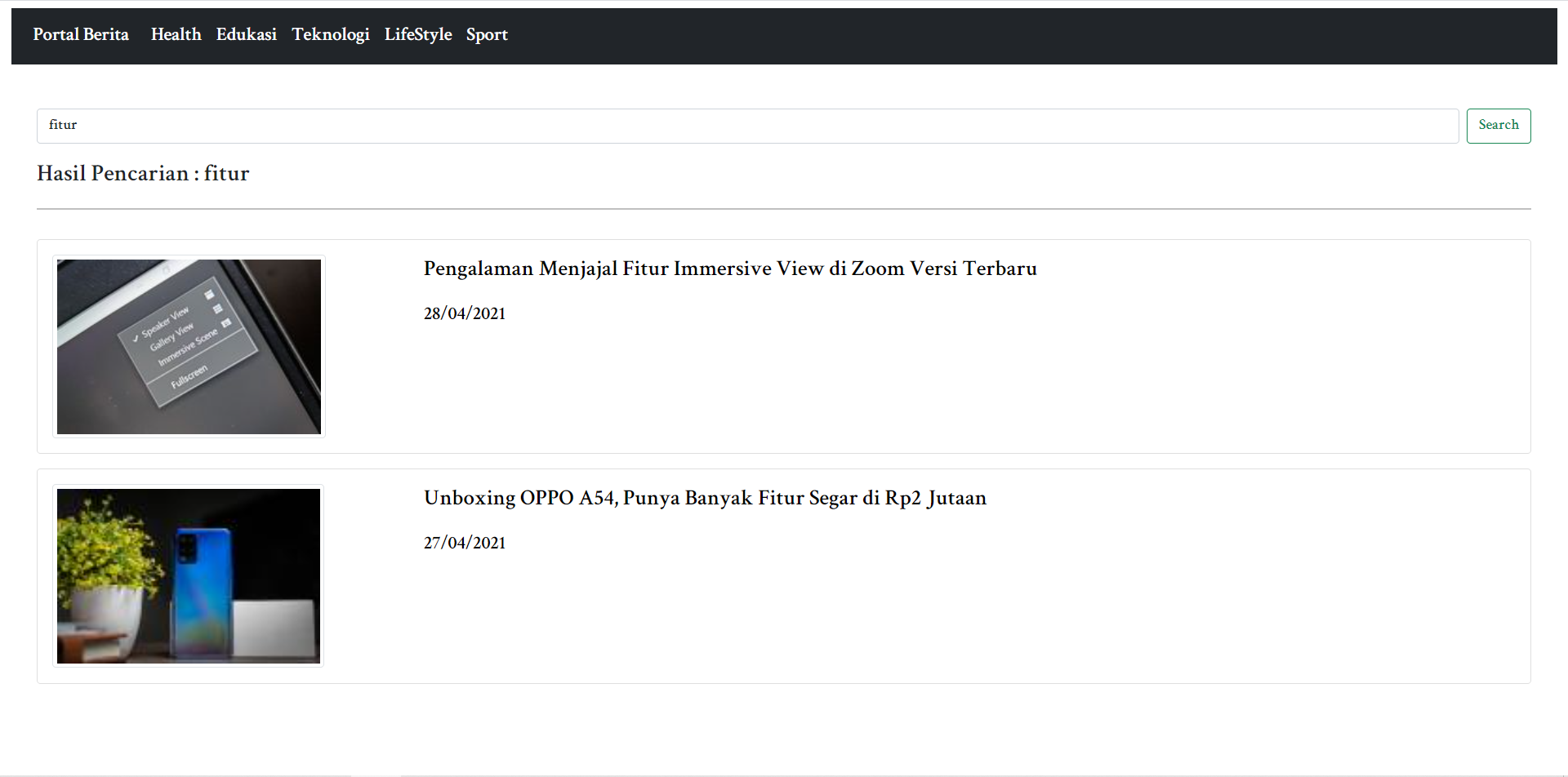
Gambar 3.6 Tampilan home

Gambar 3.6 merupakan tampilan awal saat *user* membuka *website*. Tampilan tersebut memiliki fitur dimana *user* dapat melihat semua berita dan *user* juga dapat mencari berita yang diinginkan. Tampilan utama dari *website* tersebut juga memiliki *navigation bar* yang bertujuan untuk mengkategorikan setiap berita yang ada dan sesuai dengan apa yang terdapat dalam portal berita kompas.com maupun sindonews.com. Saat salah satu pilihan dipilih oleh *user*, akan ditampilkan tampilan yang sama dengan tampilan *home*. Akan tetapi, berita yang disajikan hanya berita yang sesuai dengan pilihan kategori user.



Gambar 3.7 Detail berita

Gambar 3.7 merupakan tampilan detail berita. Tampilan tersebut dapat diakses dengan memilih berita yang diingkan oleh *user*. Detail berita akan menampilkan data-data yang berhubungan dengan berita yang dipilih seperti judul berita, tanggal berita, isi dan *tags* dimana hasil *tags* tersebut didapat dengan hasil klasifikasi yang didapat dengan menggunakan model BERT yang telah dibuat.



Gambar 3.8 Hasil search

Gambar 3.8 merupakan tampilan *search*. *Search* merupakan salah satu fitur yang berguna untuk mempermudah *user* mencari berita yang diinginkan. *Search* yang dicari akan menghasilkan tampilan yang menunjukkan berbagai berita yang sesuai dengan yang diinginkan oleh *user*. Ketika salah satu berita dipilih oleh *user*, *user* akan diperlihatkan detail dari berita tersebut.

# IMPLEMENTASI SISTEM

Pada bab ini akan dibahas tentang implementasi sistem sesuai analisa sistem dan desain sistem. Pembahasan mencakup implementasi sistem, konfigurasi *scheduler* dan aplikasi berupa *website* untuk menampilkan berita. Implementasi sistem meliputi implementasi *web scraping, preprocessing, dan* implementasi *Bidirectional Encoder Representations from Transformers* untuk melakukan klasifikasi.

## Implementasi Web Scraping

*Web scraping* ini digunakan untuk mengambil data berupa berita yang dibutuhkan dari situs penyedia berita *online* seperti kompas.com dan sindonews.com. Kompas.com dan sindonews.com memiliki struktur *website* yang berbeda, sehingga untuk melakukan *scraping* terhadap 2 *website* tersebut dilakukan metode yang berbeda. Untuk melakukan scraping terhadap kompas.com dan sindonews.com akan digunakan *library BeautifulSoup.*

Segmen Program 4.1 Import library dan inisialisasi variabel

|  |
| --- |
| import requests  import time  from bs4 import BeautifulSoup  import re  import pandas as pd  txtJudul = []  txtKategori = []  txtIsi = []  txtUrl = []  start = time.time() |

Sebelum melakukan web scraping perlu mengimport semua library yang dibutuhkan dan inisialisasi semua variabel yang diperlukan untuk mempermudah penulis untuk melakukan web scraping untuk mendapatkan url, judul, isi beserta kategori dari suatu berita. Url yang akan digunakan untuk mengambil isi dari suatu berita karena url untuk mengambil isi berita berbeda dengan url yang digunakan untuk mengambil judul berita.

Segmen Program 4.2 Fungsi scraping kompas.com

|  |
| --- |
| def scrapingKompas(kategori):  url = 'https://indeks.kompas.com/?site={}&date='.format(kategori)  for tgl in range(1,32):  #Looping per page  url1 = url+'2021-01-{}&page='.format(tgl)  #manual sdh di cek maks hny 4  for page in range(1,4):  req1 = requests.get(url1+"{}".format(page))  soup = BeautifulSoup(req1.text, 'html.parser')  items = soup.findAll('div', 'article\_\_list\_\_title')  for item in items:  txtUrl.append(item.find('a').get('href'))  txtJudul.append(item.find('a', 'article\_\_link').text)  txtKategori.append(kategori)    tUrl = "{}?page=all".format(item.find('a').get('href'))  reqUrl = requests.get(tUrl)  soupUrl = BeautifulSoup(reqUrl.text, 'html.parser')  contents = soupUrl.findAll('div', 'read\_\_content')    for content in contents:  textHapus = []  listText = content.findAll('strong')  fullText = content.text  #listText=content.find('p').find('strong')  #print(listText.text)  i = 0  for a in listText:  i+=1  if(a.text[:10])=="Baca juga:":  textHapus.append(a)  elif (a.text[:10].lower())=="kompas.com":  textHapus.append(a)    fullText = re.sub('\([Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa].\*','',fullText)    for i in textHapus:  #fullText = fullText.replace(re.sub('<[^<]+?>', '', str(i)),'')  fullText = fullText.replace(re.sub('<[^<]+?>', '', str(i)),'')  fullText = fullText.replace("“", "")  fullText = fullText.replace("'", "") #karena db kalau ada tanda '(single quote) error  txtIsi.append(fullText) |

Segmen Program 4.2 merupakan sebuah fungsi untuk melakukan *scraping* terhadap *website* kompas.com. Fungsi tersebut berguna untuk mendapatkan semua data berita yang dibutuhkan seperti url yang akan digunakan untuk mendapatkan isi berita, judul, isi berita dan kategori dari berita tersebut.

Segmen Program 4.3 fungsi scraping sindonews.com

|  |
| --- |
| def scrapingSindo(kategori):  if kategori == "edukasi":  tmp = 'https://index.sindonews.com/index/144'  elif kategori == "tekno":  tmp = 'https://index.sindonews.com/index/612'  elif kategori == "sports":  tmp = 'https://index.sindonews.com/index/10'  elif kategori == "lifestyle":  tmp = 'https://index.sindonews.com/index/154'    for tgl in range(1,32):  #url = tmp + '?t=2021-01-{}'.format(tgl)  page = 0  for i in range(1,4):  url1 = tmp + "/{}/?t=2021-01-{}".format(page,tgl)  req1 = requests.get(url1, timeout=None)  soup1 = BeautifulSoup(req1.text, 'html.parser')  items = soup1.findAll('div', 'indeks-rows')    for item in items:  txtUrl.append(item.find('div', 'indeks-title').find("a").get('href'))  txtJudul.append(item.find('div', 'indeks-title').text)  txtKategori.append(kategori)  tesIndeks.append(len(txtJudul))  tUrl = "{}?showpage=all".format(item.find('div', 'indeks-title').find("a").get('href'))  tesUrl.append(tUrl)  reqUrl = requests.get(tUrl)  soupUrl = BeautifulSoup(reqUrl.text, 'html.parser')  if kategori == 'edukasi':  contents = soupUrl.findAll('div', 'article')  elif kategori == 'tekno':  contents = soupUrl.findAll('div', 'desc-artikel-detail')  else:  contents = soupUrl.findAll('div', {"id": "content"})  for content in contents:  for br in content.findAll('br'):  br.replace\_with("\n")  fullText = ""  textHapus = []  #txtIsi.append(content.text)  if(kategori == 'edukasi'):  fullText = content.text  listText = content.findAll('strong')  fullText = re.sub('\([Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa].\*','',fullText)  fullText = re.sub('\([Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa].\*','',fullText)  i = 0  if(content.find('div', 'baca-inline')):  textHapus.append(content.find('div', 'baca-inline').text)  if(content.find('div', 'reporter')):  textHapus.append(content.find('div', 'reporter').text)  for a in content.findAll('div', 'social-embed'):  textHapus.append(a)  for a in listText:  i+=1  if i==1:  textHapus.append(a.text + ' - ')  else:  #print(a.text[1:10].lower(), a.text[-1])  if a.text[1:10].lower()=='baca juga':  #print(a.findNext('strong').text, tesIndeks[-1])  textHapus.append(a.text)  textHapus.append(a.text + a.findNext("a").text)  elif(kategori == 'sports'):  fullText = content.text  listText = content.findAll('strong')  fullText = re.sub('\([Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa].\*','',fullText)  fullText = re.sub('\([Ii][Kk][Uu][Tt][Ii] [Ss][Uu][Rr][Vv][Ee][Ii]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Ii][Kk][Uu][Tt][Ii] [Ss][Uu][Rr][Vv][Ee][Ii].\*','',fullText)  fullText = re.sub('\([Ll][Ii][Hh][Aa][Tt] [Gg][Rr][Aa][Ff][Ii][Ss]([\s\S]\*?)\)','',fullText)    fullText = re.sub('\([Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa].\*','',fullText)  i = 0  if(content.find('div', 'baca-inline')):  textHapus.append(content.find('div', 'baca-inline').text)  if(content.find('div', 'editor')):  textHapus.append(content.find('div', 'editor').text)  #listSurvey = content.findAll('br')  #textHapus+=content.findAll('div', 'box-outlink')  if(content.find('div', 'box-outlink')):  textHapus.append(content.find('div','box-outlink').text)  """for a in listSurvey:  if a.findNext('strong'):  if(a.findNext('strong').text[1:13]=='Ikuti Survei'):    textHapus.append(a.findNext('strong').text)"""  for a in content.findAll('div', 'social-embed'):  textHapus.append(a)  for a in listText:  i+=1  if i==1:  textHapus.append(a.text + ' - ')  else:  if a.text[1:11].lower()=='(baca juga':  textHapus.append(a.text)  #print(a.text, tesIndeks[-1])  #print(a.text, tesIndeks[-1])  elif a.text[1:10].lower()=='baca juga':  textHapus.append(a.text)  #print(a.text, tesIndeks[-1])  #print(a.text, tesIndeks[-1])  elif a.text[0:9].lower()=='baca juga':  textHapus.append(a.text)  #print(a.text, tesIndeks[-1])  elif a.text[0:14].lower()=='(lihat grafis:':  textHapus.append(a.text)  #print(a.text, tesIndeks[-1])    #print(a.text, tesIndeks[-1])  #print(a.text)  elif(kategori == 'tekno'):  fullText = content.text  fullText = re.sub('\([Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa].\*','',fullText)    fullText = re.sub('\([Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa].\*','',fullText)    listText = content.findAll('strong')  if(content.find('div', 'warp-baca-juga')):  textHapus.append(content.find('div', 'warp-baca-juga').text)    if(content.find('div', 'editor')):  textHapus.append(content.find('div', 'editor').text)    #if(content.find('div', 'box-outlink')):  # textHapus.append(content.find('div','box-outlink').text)  for a in content.findAll('div', 'social-embed'):  textHapus.append(a)  i = 0  for a in listText:  i+=1  if i==1:  textHapus.append(a.text + ' - ')    elif(kategori == 'lifestyle'):  fullText = content.text  #fullText = re.sub('\([Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa][^)]\*\)','',fullText)  #fullText = re.sub('[Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa]([\s\S]\*?)(<br>)','',fullText)  fullText = re.sub('\([Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Bb][Aa][Cc][Aa] [Jj][Uu][Gg][Aa].\*','',fullText)  fullText = re.sub('\([Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa]([\s\S]\*?)\)','',fullText)  fullText = re.sub('[Ll][Ii][Hh][Aa][Tt] [Jj][Uu][Gg][Aa].\*','',fullText)  #print(fullText)  if(content.find('div', 'baca-inline')):  textHapus.append(content.find('div', 'baca-inline').text)  if(content.find('div', 'editor')):  textHapus.append(content.find('div', 'editor').text)  if(content.find('div', 'box-outlink')):  textHapus.append(content.find('div','box-outlink').text)  listText = content.findAll('strong')  for a in content.findAll('div', 'social-embed'):  textHapus.append(a)  i = 0  for a in listText:  i+=1  if i==1:  textHapus.append(a.text + ' - ')  listText = content.findAll('strong')  listText += content.findAll('em')  for a in listText:  i+=1  if str(a.text[1:10].lower())=='baca juga' or str(a.text[0:9].lower())=='baca juga':  textHapus.append(a)  elif i == 1:  textHapus.append(a.text+' - ')  for i in textHapus:  fullText = fullText.replace(re.sub('<[^<]+?>', '', str(i)),'')    fullText = fullText.replace("“", "")  fullText = fullText.replace("'", "") #karena db kalau ada tanda '(single quote) error  txtIsi.append(fullText)    page+=15 |

Segmen Program 4.3 merupakan sebuah fungsi untuk melakukan *scraping* terhadap *website* sindonews.com. Fungsi tersebut berguna untuk mendapatkan semua data berita yang dibutuhkan seperti url yang akan digunakan untuk mendapatkan isi berita, judul, isi berita dan kategori dari berita tersebut. Setelah melakukan *scraping* terhadap kompas.com dan sindonews.com, berita yang didapatkan akan diubah ke format .csv untuk digunakan lebih lanjut. Segmen program 4.4 merupakan *source code* untuk mengubah semua berita yang didapatkan kedalam format .csv.

Segmen Program 4.4 Convert to csv

|  |
| --- |
| scrapingKompas("edukasi")  scrapingKompas("tekno")  scrapingKompas("sports")  scrapingKompas("health")  scrapingKompas("lifestyle")  scrapingSindo('edukasi')  scrapingSindo('sports')  scrapingSindo('tekno')  scrapingSindo('lifestyle')    df = pd.DataFrame({  "judul": txtJudul,  "url": txtUrl,  "isi": txtIsi,  "kategori": txtKategori  })  df.to\_csv('outputBerita.csv', index=False, encoding='utf-8')  end = time.time()  hours, rem = divmod(end-start, 3600)  minutes, seconds = divmod(rem, 60)  print("{:0>2}:{:0>2}:{:05.2f}".format(int(hours),int(minutes),seconds)) |



Gambar 4.1 Output segmen program 4.4

## Implementasi Preprocessing

Sebelum melakukan klasifikasi, tahap yang perlu dilakukan adalah preprocessing text. Preprocessing text bertujuan untuk membersihkan data dari noise. Preprocessing meliputi filter string, stopword removal, stemming.

Segmen Program 4.5 Fungsi preprocessing text

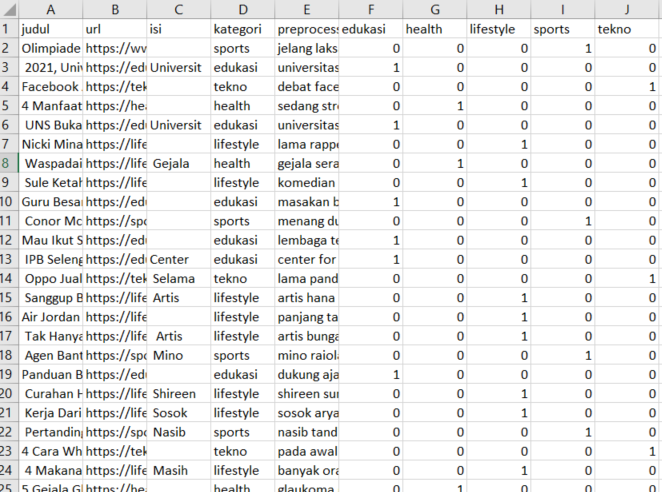
|  |
| --- |
| def preprocessing\_text(text):  encoded\_string = text.encode("ascii", "ignore") #remove asci  text = encoded\_string.decode() #remove asci  text = text.lower() #lowercase    text = ''.join([i for i in text if not i.isdigit()]) #remove number  punctuations = '''!()-[]{};:'"\,<>./?@#$%^&\*“”‘’\_~+=|\t\n'''  for char in text:  if char in punctuations:  text = text.replace(char, "")  text = re.sub(r'http\S+', '', text) #remove url    factory = StopWordRemoverFactory()  stopword = factory.create\_stop\_word\_remover()  text = stopword.remove(text)    factory = StemmerFactory()  stemmer = factory.create\_stemmer()  text = stemmer.stem(text)    return text |

Filter String merupakan proses pembersihan dimana hanya menerima huruf dan semua yang tidak mengandung huruf akan dihapus seperti tanda baca, angka, dan special character dalam tahapan ini juga merubah semua huruf besar menjadi huruf kecil (Lowercase text). Proses ini juga akan menghapus url karena tidak diperlukan untuk melakukan klasifikasi. Case Folding merupakan proses dimana huruf besar yang ada pada text akan diubah menjadi huruf kecil. Stopword removal merupakan proses dimana menghilangkan kata berulang yang tidak memiliki arti. Stemming merupakan proses dimana kata akan diubah menjadi kata baku (root word). Proses stopword removal dan stemming akan dilakukan dengan bantuan library Sastrawi.

Segmen Program 4.6 Ubah menjadi list kategori dan Convert ke csv

|  |
| --- |
| start = time.time()  df['preprocessing\_text'] = df.isi.apply(preprocessing\_text)  dummies = pd.get\_dummies(df.kategori)  df = pd.concat([df, dummies], axis='columns')  end = time.time()  df.to\_csv('datasetBeritaFixWithDummies.csv', index=False, encoding='utf-8')  #df.to\_csv('train\_berita.csv', index=False, encoding='utf-8')  end = time.time()  hours, rem = divmod(end-start, 3600)  minutes, seconds = divmod(rem, 60)  print("{:0>2}:{:0>2}:{:05.2f}".format(int(hours),int(minutes),seconds)) |

Setelah melakukan Segmen Program 4.5, langkah selanjutnya adalah mengubah kategori kedalam bentuk list untuk memudahkan saat *training*. Langkah terakhir yang dilakukan adalah menyimpannya kedalam format .csv agar memudahkan untuk diolah lebih lanjut karena proses *stemming* membutuhkan waktu yang lama. Proses tersebut dapat dilihat pada Segmen Program 4.6. Output Segmen Program 4.6 dapat dilihat pada Gambar 4.2.



Gambar 4.2 Output Segmen Program 4.6

## Implementasi Klasifikasi

Setelah melakukan *preprocessing text*, langkah selanjutnya yang dilakukan adalah melakukan klasifikasi dengan metode *Bidirectional Encoder Representations from Transformers*. Tahapan klasifikasi dilakukan dengan menggunakan *library* Pytorch dan dijalankan dengan menggunakan Google Collab. Training akan dijalankan dengan menggunakan GPU. GPU berguna untuk mengoptimalkan proses komputasi dalam membangun model deep learning. Oleh karena itu, diperlukan pengecekan untuk mengetahui apakah GPU tersedia atau tidak. Segmen Program 4.7 merupakan *source code* untuk mengecek ketersediaan GPU.

Segmen Program 4.7 Cek GPU

|  |
| --- |
| if torch.cuda.is\_available():    device = torch.device('cuda')    print('there are %d GPU(s) available.' % torch.cuda.device\_count())    print('we will use the GPU: ', torch.cuda.get\_device\_name(0))  else:    print("No GPU available, using the CPU instead")    device = torch.device("cpu")    print(device) |

Setelah memastikan GPU tersedia, langkah berikutnya yang harus dilakukan adalah menghubungkan Google Collab dengan Google Drive. Hal ini bertujuan untuk memudahkan penulis dalam membaca data tanpa melakukan inisialisasi data ke dalam Google Collab berulang kali. Untuk menghubungkan Google Collab dengan Google Drive dapat dilihat pada Segmen Program 4.8.

Segmen Program 4.8 Menghubungkan ke Google Drive

|  |
| --- |
| from google.colab import drive  drive.mount("/content/drive/") |

### Pengolahan Dataset

Dataset yang digunakan adalah berita yang didapat dari hasil scraping portal berita yaitu kompas.com dan sindonews.com. Data yang didapat juga sudah dilakukan *preprocessing* *text* yang bertujuan untuk membersihkan data dari *noise*. Untuk memudahkan proses *training* dan *testing*, kategori yang sebelumnya masih belum merupakan sebuah list perlu diubah terlebih dahulu.

Segmen Program 4.9 Mengubah kategori menjadi list

|  |
| --- |
| #train\_df=pd.read\_csv('/content/drive/MyDrive/Skripsi/FIX/datasetBeritaFixWithDummies.csv')  train\_df = pd.read\_csv('/content/drive/MyDrive/Skripsi/FIX/train\_df.csv')  df = pd.read\_csv('/content/drive/MyDrive/Skripsi/FIX/databaru.csv') #DATA BARU  class\_names = ['edukasi', 'tekno', 'sports', 'health', 'lifestyle']  #test\_df['list\_kategori'] = list(test\_df[class\_names].values)  train\_df['list\_kategori'] = list(train\_df[class\_names].values)  train\_df=train\_df.drop(columns=class\_names)  #test\_df=test\_df.drop(columns=class\_names) |

Sebelum mengubah menjadi data menjadi input yang dapat dibaca oleh BERT, dilakukan konfigurasi *hyperparameter* terlebih dahulu. Segmen Program 4.10 merupakan *source code* untuk menginisialisasi *hyperparameter* yang dibutuhkan.

Segmen Program 4.10 Inisialisasi hyperparameter

|  |
| --- |
| EPOCHS = 4 #BERT Recommend using 2-4 EPOCH  LEARNING\_RATE = 2e-5 #5e-5, 4e-5, 3e-5, 2e-5, 1e-5  DROPOUT = 0.1  BATCH\_SIZE = 16  PRE\_TRAINED\_MODEL\_BAHASA =  'indobenchmark/indobert-base-p1'  tokenizer = BertTokenizer.from\_pretrained(PRE\_TRAINED\_MODEL\_BAHASA)  MAX\_LEN = 320 |

Tokenizer menggunakan *library* BERT yang dapat dilihat di Hugging Face dan model yang digunakan adalah indobenchmark/indobert-base-p1 karena model ini merupakan model untuk Bahasa Indonesia. Model tersebut memiliki *base* yang sama dengan model lain dari BERT seperti bert-base-multilingual-cased, bert-base-multilingual-uncased, dll. Dataset Bahasa Indonesia yang digunakan dalam melatih model ini terdiri dari sekitar 4 miliar kata, dengan sekitar 250 juta kalimat diantaranya mencakup Bahasa Indonesia formal dan Bahasa sehari-hari (Wilie et al., 2020). Model ini juga dapat digunakan untuk menyelesaikan tugas nlp lainnya.

Untuk mengetahui kegunaan variabeel *hyperparameter* yang diinisialisasi, daftar beserta keterangan dapat dilihat pada Tabel 4.1.

Tabel 4.1 Keterangan Hyperparameter

|  |  |  |
| --- | --- | --- |
| No | Variabel | Keterangan |
| 1 | EPOCHS | Jumlah iterasi yang akan dilakukan untuk melakukan training model. |
| 2 | LEARNING\_RATE | Parameter yang menghitung nilai bobot pada proses training. Semakin rendah, semakin lama training bekerja sedangkan semakin besar nilai *learning rate* maka semakin cepat training bekerja namun tidak optimal. |
| 3 | DROPOUT | Layer yang digunakan untuk mencegah suatu model mengalami *overfit.* Dimana beberapa neuron akan dipilih secara acak untuk tidak dipakai selama pelatihan berdasarkan persentase yang diinisialisasi oleh pembuat. |
| 4 | BATCH\_SIZE | Banyaknya data yang dapat diproses oleh komputer untuk disebarkan ke Neural Network. Contoh batch size=16, jumlah dataset 160 maka akan menggunakan 16 data pertama untuk dapat diproses terlebih dahulu. Proses ini akan berlanjut hingga semua dataset telah diambil. Banyaknya iterasi 160/16 sehingga akan dilakukan hingga data kesepuluh. Iterasi ini akan dijalankan untuk 1 epoch. |
| 5 | TOKENIZER | Untuk mengubah kalimat menjadi token |
| 6 | MAX\_LEN | Untuk menentukan panjang kalimat yang diteruskan. Panjang kalimat ini ditentukan berdasarkan rata-rata jumlah token yang ada pada suatu dataset. |

BERT memiliki maksimal panjang 512 token, Oleh karena itu, diperlukan rata-rata dari panjang token yang ada di dalam dataset untuk menjadi max\_len. Untuk mengetahui panjang token, dapat dilihat pada Segmen Program 4.11.

Segmen Program 4.11 Menentukan rata-rata panjang token

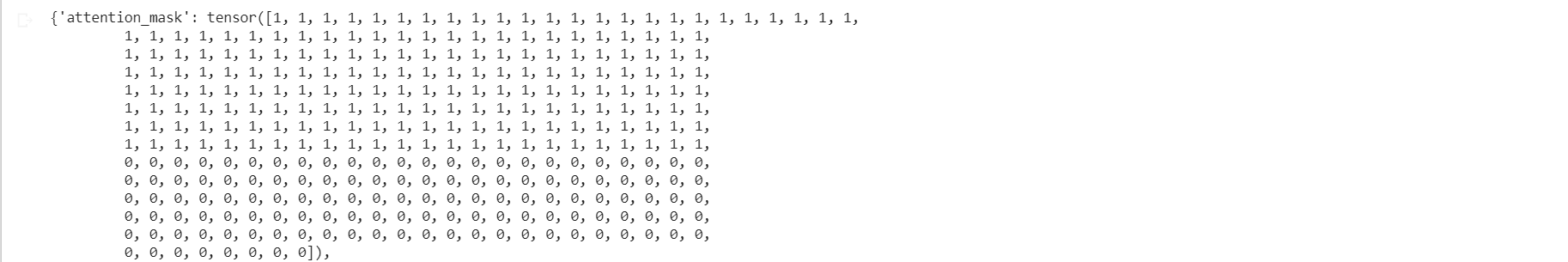
|  |
| --- |
| import statistics  sent\_length = []  # For every sentence...  for sent in train\_df.preprocessing\_text:      input\_ids = tokenizer.encode(sent, add\_special\_tokens=True)      sent\_length.append(len(input\_ids))  print('Average length = ', sum(sent\_length)/len(sent\_length)) |

Langkah berikutnya yang perlu dilakukan adalah mengubah dataset menjadi data yang dapat dibaca oleh BERT. Oleh karena itu, pada skripsi ini akan dibuat sebuah kelas untuk mengubah dataset yang ada menjadi input representasi yang dapat dibaca oleh BERT.

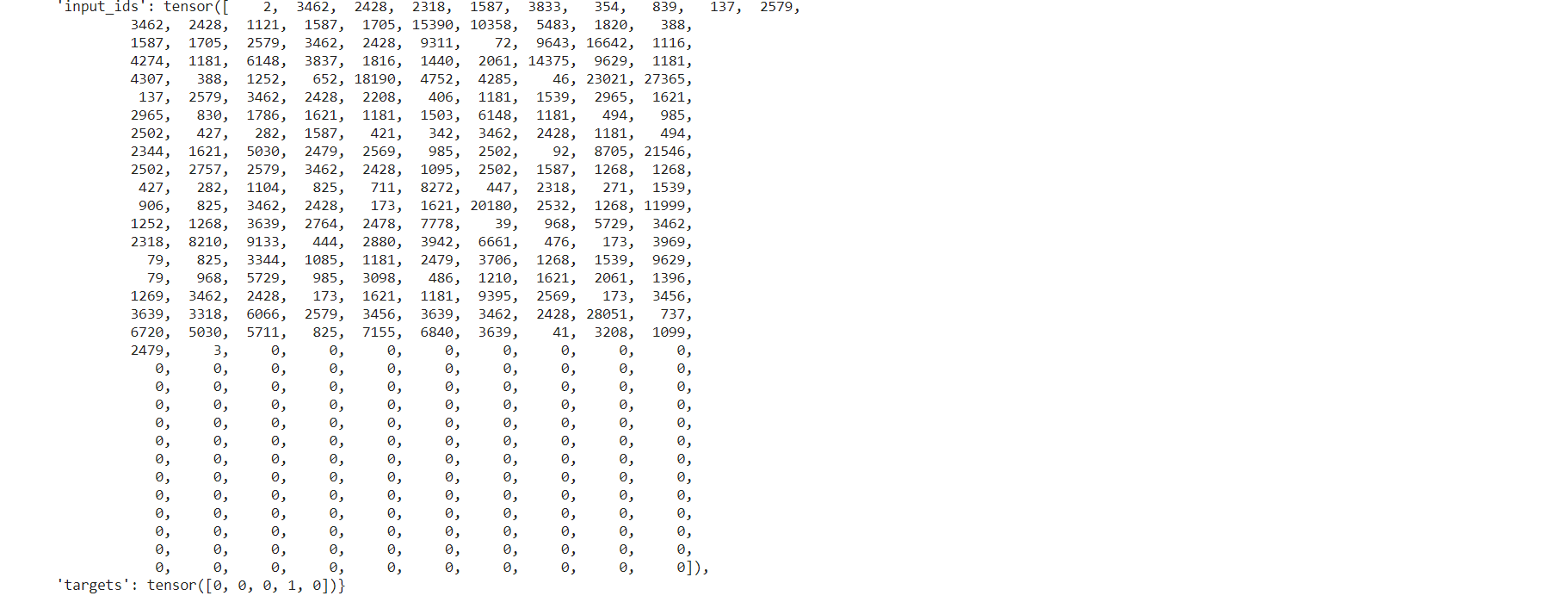
Segmen Program 4.12 Input Representation BERT

|  |
| --- |
| class DatasetBerita(Dataset):    def \_\_init\_\_(self, text, kategori, tokenizer, max\_len):      self.isi = text      self.category = kategori      self.tokenizer = tokenizer      self.max\_len = max\_len    def \_\_len\_\_(self):      return len(self.isi)    def \_\_getitem\_\_(self, item):      isi\_berita = str(self.isi[item])      kategori = self.category[item]      encoding = self.tokenizer.encode\_plus(        isi\_berita,        add\_special\_tokens=True,        max\_length=self.max\_len,        return\_token\_type\_ids=False,        padding='max\_length',        return\_attention\_mask=True,        truncation=True,        return\_tensors='pt',      )      return {        'input\_ids': encoding['input\_ids'].flatten(),        'attention\_mask': encoding['attention\_mask'].flatten(),        'targets': torch.tensor(kategori, dtype=torch.long)      } |

BERT pertama mengubah kalimat menjadi token dan menambahkan token [CLS] di awal kalimat, [SEP] di akhir kalimat dan menambahkan token [PAD] setelah token [SEP] sebanyak sisa dari panjang max\_len. Setelah menambahkan ketiga token tersebut, BERT mengubah setiap token menjadi id berupa posisi letak vocabulary file yang sudah diinisalisasi oleh pembuat model.



Gambar 4.3 Attention Mask



Gambar 4.4 Input Ids dan Kategori

Untuk keterangan lebih lanjut mengenai tentang *input representation* BERT maka dapat dilihat pada tabel

Tabel 4.2 Input Representations BERT

|  |  |  |
| --- | --- | --- |
| No | Output | Keterangan |
| 1 | Input\_ids | id dari sebuah token yang sudah diinialisasi sebelumnya oleh pembuat model. Input\_ids juga terdapat id special token berupa [CLS], [SEP] dan [PAD] |
| 2 | Attention\_mask | Berupa integer 0 dan 1. 0 merupakan padding sedangkan 1 merupakan kata sebenarnya (selain padding). Attention mask berguna untuk |
| 3 | Targets | List kategori. Contoh: edukasi maka targets bernilai [1, 0, 0, 0, 0] |

Setelah mengubah dataset menjadi *input* yang dapat dibaca oleh BERT, akan dilakukan pembagian data menjadi data training dan testing dan meneruskannya ke dalam DataLoader. DataLoader berguna untuk merandom data pada setiap epoch untuk mencegah model dari *overfit* dan membantu mempercepat pengambilan data.

Segmen Program 4.13 Split testing data dan pembuatan dataloader

|  |
| --- |
| RANDOM\_SEED = 42  train\_df, test\_df = train\_test\_split(    train\_df,    test\_size=0.2, # train 80% val 20%    random\_state=RANDOM\_SEED  )  train\_data\_loader = DataLoader(DatasetBerita(      train\_df.preprocessing\_text.to\_numpy(),      kategori=train\_df.list\_kategori.to\_numpy(),      tokenizer=tokenizer,max\_len=MAX\_LEN), batch\_size=BATCH\_SIZE, shuffle=True, num\_workers=4)  test\_data\_loader = DataLoader(DatasetBerita(      test\_df.preprocessing\_text.to\_numpy(),      kategori=test\_df.list\_kategori.to\_numpy(),      tokenizer=tokenizer,max\_len=MAX\_LEN), batch\_size=BATCH\_SIZE, shuffle=True, num\_workers=4) |

### Bidirectional Encoder Representations from Transformers

Pada subbab ini akan dijelaskan bagaimana cara membuat model untuk melakukan klasifikasi, *training*, dan *testing*. Dalam pembuatan model, model yang dibangun menggunakan layer yang sama dengan function BertForSequenceClassification. Namun, penulis tidak memakai function yang disediakan oleh Huggingface karena tidak dapat mengkonfigurasi nilai *dropout layer* secara manual. Pembuatan model dapat dilihat pada Segmen Program 4.14.

Segmen Program 4.14 model BERT

|  |
| --- |
| class Klasifikasi(nn.Module):    def \_\_init\_\_(self, n\_classes, dropout\_value=0.2):      super(Klasifikasi, self).\_\_init\_\_()      self.bert = BertModel.from\_pretrained(PRE\_TRAINED\_MODEL\_BAHASA)      self.drop = nn.Dropout(p=dropout\_value)      self.out = nn.Linear(self.bert.config.hidden\_size, n\_classes)    def forward(self, input\_ids, attention\_mask):      \_, pooled\_output = self.bert(        input\_ids=input\_ids,        attention\_mask=attention\_mask,        return\_dict=False      )      output = self.drop(pooled\_output)      return self.out(pooled\_output) |

Segmen Program 4.15 Konfigurasi optimizer, scheduler

|  |
| --- |
| optimizer = AdamW(model.parameters(), lr=LEARNING\_RATE, correct\_bias=False)  total\_steps = len(train\_data\_loader) \* EPOCHS  scheduler = get\_linear\_schedule\_with\_warmup(    optimizer,    num\_warmup\_steps=0,    num\_training\_steps=total\_steps  )  loss\_fn = nn.CrossEntropyLoss().to(device) |

*Optimizer* yang digunakan adalah Adam. *Optimizer* ini berguna untuk meminimalisir kesalahan pada saat *training*. *Loss* yang digunakan adalah cross entropy (untuk menghitung *loss* tugas klasifikasi). *Loss* digunakan untuk menghitung loss yang terjadi pada saat *training* *model.*

Segmen Program 4.16 Fungsi training

|  |
| --- |
| def train\_epoch( model, data\_loader, loss\_fn, optimizer, device, scheduler):    model = model.train()    losses = []    pred\_labels = []    true\_labels = []    for d in data\_loader:      input\_ids = d["input\_ids"].to(device)      attention\_mask = d["attention\_mask"].to(device)      targets = d["targets"].to(device)      outputs = model(        input\_ids=input\_ids,        attention\_mask=attention\_mask      )      loss = loss\_fn(outputs, torch.max(targets.float(), 1)[1])      outputs = F.softmax(outputs,dim=1)      b\_probs = outputs.detach().cpu().numpy()      b\_targets = targets.detach().cpu().numpy()      true\_labels.append(b\_targets)      pred\_labels.append(b\_probs)      losses.append(loss.item())      loss.backward()      nn.utils.clip\_grad\_norm\_(model.parameters(), max\_norm=1.0)      optimizer.step()      scheduler.step()      optimizer.zero\_grad()      pred\_labels = [item for sublist in pred\_labels for item in sublist]    true\_labels = [item for sublist in true\_labels for item in sublist]      threshold = 0.50    pred\_bools = [pl>threshold for pl in pred\_labels]    true\_bools = [tl==1 for tl in true\_labels]    return accuracy\_score(true\_bools, pred\_bools), np.mean(losses) |

Segmen Program 4.17 Fungsi evaluasi

|  |
| --- |
| def eval\_model(model, data\_loader, loss\_fn, device):    model = model.eval()    losses = []    pred\_labels = []    true\_labels = []    with torch.no\_grad():      for d in data\_loader:        input\_ids = d["input\_ids"].to(device)        attention\_mask = d["attention\_mask"].to(device)        targets = d["targets"].to(device)        outputs = model(          input\_ids=input\_ids,          attention\_mask=attention\_mask        )        loss = loss\_fn(outputs, torch.max(targets.float(), 1)[1])        outputs = F.softmax(outputs,dim=1)        b\_probs = outputs.detach().cpu().numpy()        b\_targets = targets.detach().cpu().numpy()        true\_labels.append(b\_targets)        pred\_labels.append(b\_probs)        losses.append(loss.item())    pred\_labels = [item for sublist in pred\_labels for item in sublist]    true\_labels = [item for sublist in true\_labels for item in sublist]    threshold = 0.50    pred\_bools = [pl>threshold for pl in pred\_labels]    true\_bools = [tl==1 for tl in true\_labels]    return accuracy\_score(true\_bools, pred\_bools), np.mean(losses) |

Dalam melakukan *training* maupun evaluasi, penulis menggunakan *layer activation* softmax. *Layer* tersebut bertujuan untuk mendapatkan probabilitas di setiap kelas. *Training* bertujuan untuk melatih model yang dibangun, sedangkan evaluasi bertujuan untuk mengevaluasi model yang dibangun. Untuk melakukan *training* maupun *testing*, fungsi ini akan dijalankan pada setiap epoch. Model terbaik nantinya akan disimpan dengan format .bin agar dapat digunakan pada saat memprediksi data baru.

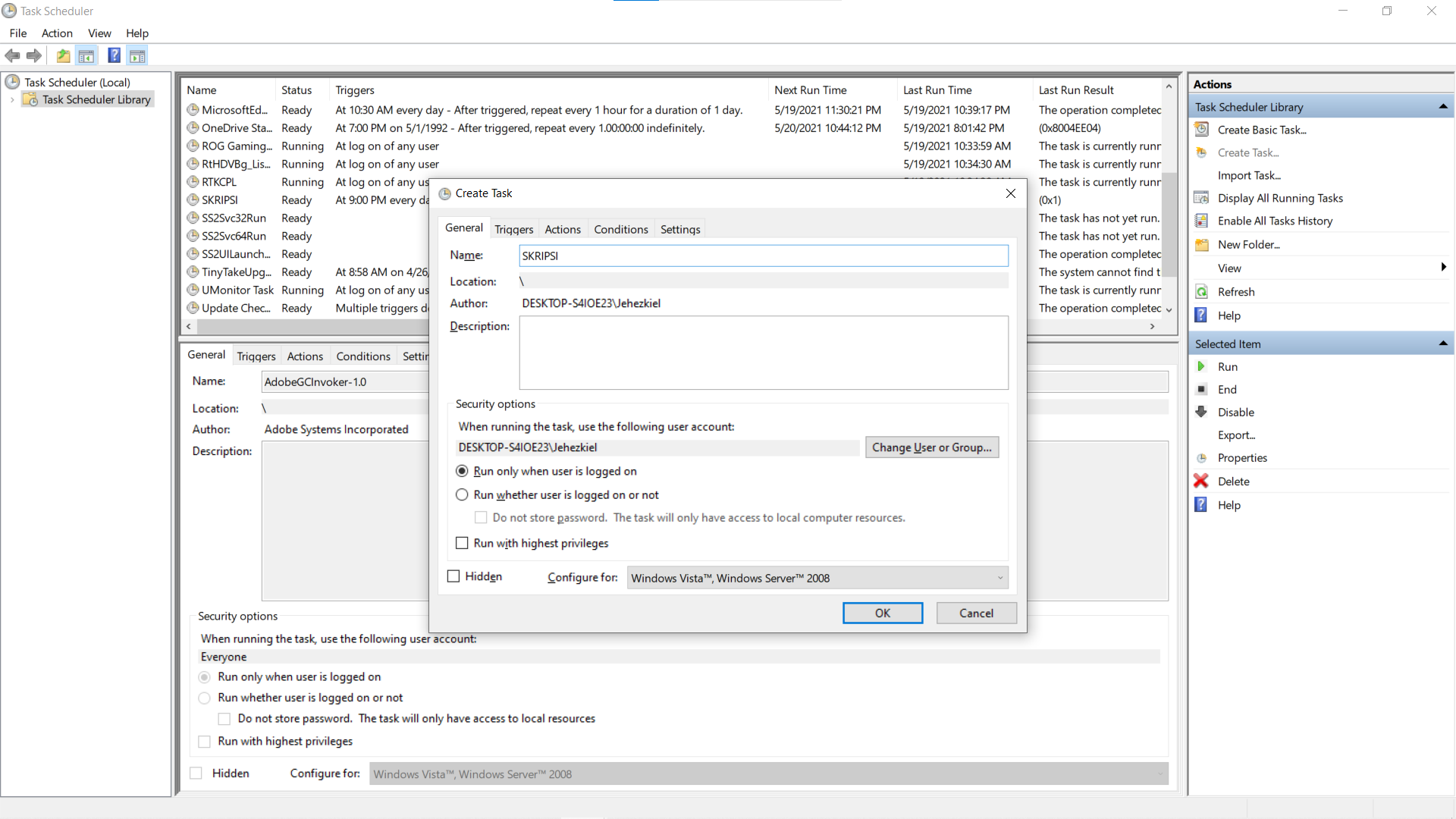
Segmen Program 4.18 Menjalankan fungsi training dan testing

|  |
| --- |
| %%time  print(f'softmax\_layer\_{LEARNING\_RATE}\_{DROPOUT}\_{BATCH\_SIZE}\_{EPOCHS}')  history = defaultdict(list)  best\_accuracy = 0  meanTrainAcc = []  meanTestAcc = []  for epoch in range(EPOCHS):    # Measure how long the training epoch takes.    t0 = time.time()    print(f'Epoch {epoch + 1}/{EPOCHS}')    print('-' \* 10)    train\_acc, train\_loss= train\_epoch(      model,      train\_data\_loader,      loss\_fn,      optimizer,      device,      scheduler    )    print(f'Train loss {train\_loss} accuracy {train\_acc}')    meanTrainAcc.append(train\_acc)    test\_acc, test\_loss= eval\_model(      model,      test\_data\_loader,      loss\_fn,      device    )    print(f'Val   loss {test\_loss} accuracy {test\_acc}')    meanTestAcc.append(test\_acc)    history['train\_acc'].append(train\_acc)    history['train\_loss'].append(train\_loss)    history['test\_acc'].append(test\_acc)    history['test\_loss'].append(test\_loss)    elapsed = format\_time(time.time() - t0)    print(f'Time : {elapsed}')    if(epoch + 1 == EPOCHS):      print(f'Rata-rata akurasi train {np.mean(meanTrainAcc)}')      print(f'Rata-rata akurasi test {np.mean(meanTestAcc)}')    if test\_acc > best\_accuracy:      torch.save(model.state\_dict(), path)      best\_accuracy = test\_acc |

## Konfigurasi Scheduler

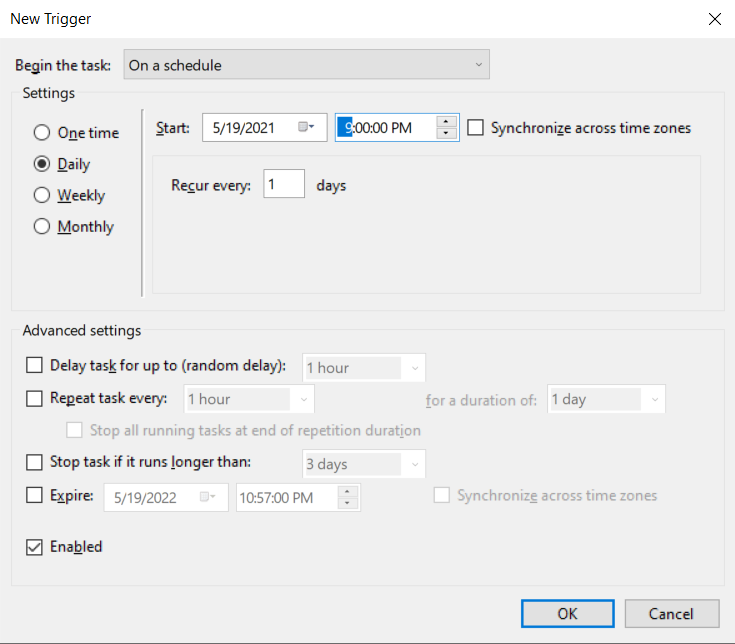
Setelah membangun model untuk melakukan klasifikasi, tahap selanjutnya yang perlu dilakukan adalah melakukan konfigurasi *scheduler.* Pada penelitian ini, *scheduler* berperan untuk menjalankan *python script* setiap hari untuk memasukkan data kedalam database agar tidak ada data yang kembar. Tahapan untuk melakukan *scheduler* dengan menggunakan *windows scheduler* adalah sebagai berikut:

* Membuat task baru dan memberi nama sesuai dengan yang diinginkan



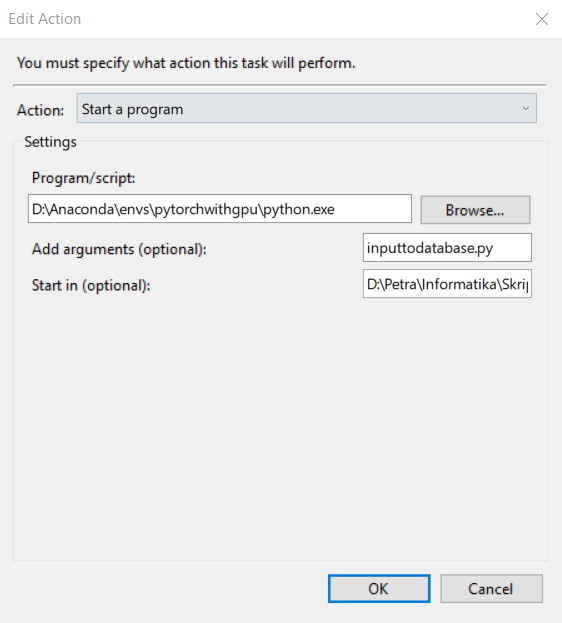
Gambar 4.5 Membuat task scheduler

* Membuat trigger baru dan atur sesuai dengan yang diinginkan (mengatur tugas yang diinginkan untuk dijalankan)



Gambar 4.6 Membuat trigger

* Tambahkan aksi yang ingin dilakukan (*add* python script). Program/script diisi dengan letak dimana python disimpan. *Add argument* diisi dengan *python script* yang ingin dijalankan. *Start in* diisi dengan letak awal dimana *python script* berada.



Gambar 4.7 Membuat aksi yang ingin dilakukan

### Prediksi berita baru

Pada subbab ini akan dijelaskan bagaimana cara memprediksi berita baru yang didapat dari portal berita kompas.com dan sindonews.com kemudian memasukkannya kedalam database. Hal pertama yang perlu dilakukan adalah *scraping* berita dan dilakukan *preprocessing* seperti sebelumnya. Langkah berikutnya adalah dengan *load* model yang telah dibuat.

Segmen Program 4.19 Load model yang telah dibuat

|  |
| --- |
| checkpoint= torch.load('/content/drive/MyDrive/Skripsi/FIX/pengujian\_model\_fix/indobert-base-p1\_1e-05\_0.2\_16\_2.bin', map\_location=torch.device('cpu'))  model = Klasifikasi(len(class\_names), DROPOUT)  optimizer = AdamW(model.parameters(), lr=LEARNING\_RATE, correct\_bias=False)  model.load\_state\_dict(checkpoint['model\_state\_dict'])  optimizer.load\_state\_dict(checkpoint['optimizer\_state\_dict']) |

Segmen Program 4.20 merupakan *source code* lanjutan yang berfokus untuk memprediksi data baru yang nantinya akan dimasukkan kedalam *database*.

Segmen Program 4.20 Fungsi untuk melakukan prediksi suatu berita

|  |
| --- |
| def PredictText(model, text):  model = model.eval()  temp\_labels = []  encoded\_review = tokenizer.encode\_plus(  text,  max\_length=MAX\_LEN,  add\_special\_tokens=True,  return\_token\_type\_ids=False,  padding='max\_length',  return\_attention\_mask=True,  truncation=True,  return\_tensors='pt',  )  input\_ids = encoded\_review['input\_ids']  attention\_mask = encoded\_review['attention\_mask']  output = model(input\_ids, attention\_mask)  prob = torch.softmax(output, dim=1)  y\_pred\_prob = prob.detach().cpu().numpy()  text = []  for a in range(len(y\_pred\_prob)):  indeks = 0  for b in range(len(y\_pred\_prob[a])):  if(y\_pred\_prob[a][b]>0.5):  text.append(class\_names[b])  return text |

Setelah mendapatkan semua data yang dibutuhkan beserta hasil prediksi, semua data tersebut akan dimasukkan kedalam database.

Segmen Program 4.21 Insert to database

|  |
| --- |
| try:  mydb = mysql.connector.connect(  host="localhost",  user="root",  password="",  database="skripsi"  )  mycursor = mydb.cursor()  for a in range(len(judul)):  mycursor.execute("INSERT into `news` (`tanggal`,`gambar`,`judul`,`isi`,`kategori`,`sumber`, `prediksi`) values('%s', '%s', '%s', '%s', '%s', '%s', '%s')" % (tanggal[a],gambar[a],judul[a], isiberita[a], kategori[a], sumber[a], toString(pred\_label[a])))  mydb.commit()  except mysql.connector.Error as err:  print(err)  else:  mydb.close() |

## Aplikasi

Pada subbab ini akan dijelaskan cara untuk menghubungkan dengan database dan sql untuk menampilkan berita. Aplikasi berupa website dan menggunakan Bahasa PHP dan menggunakan Bootstrap untuk tampilan website. Cara menghubungkan kedalam database dapat dilihat pada Segmen Program 4.21.

Segmen Program 4.22 Menghubungkan dengan database

|  |
| --- |
| <?php  $hostname = "localhost"; $username = "root";  $password = "";  $database = "skripsi";  $con = mysqli\_connect ($hostname, $username, $password, $database);  ?> |

### Navbar

Navbar merupakan *navigation bar* yang berada di atas (*header*) berupa menu. Navbar memudahkan user untuk mencari fitur yang diinginkan oleh *user* yang ada pada sebuah website. Pada setiap tampilan yang disediakan oleh sebuah website, terkadang terdapat navbar karena navbar merupakan komponen utama dari sebuah website. Pada pembuatan aplikasi ini, juga terdapat navbar yang dapat dilihat pada Segmen Program 4.23.

Segmen Program 4.23 Navbar

|  |
| --- |
| <nav class="navbar navbar-expand-lg navbar-dark bg-dark">  <div class="container-fluid">  <a class="navbar-brand" href="index.php">Portal Berita</a>  <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">  <span class="navbar-toggler-icon"></span>  </button>  <div class="collapse navbar-collapse" id="navbarSupportedContent">  <ul class="navbar-nav me-auto mb-2 mb-lg-0">  <li class="nav-item"><a class="nav-link active" aria-current="page" href="health.php">Health</a></li>  <li class="nav-item"><a class="nav-link active" href="edukasi.php">Edukasi</a></li>  <li class="nav-item"><a class="nav-link active" href="teknologi.php">Teknologi</a></li>  <li class="nav-item"><a class="nav-link active" href="lifestyle.php">LifeStyle</a></li>  <li class="nav-item"><a class="nav-link active" href="sport.php">Sport</a></li>  </ul>  </div>  </div>  </nav> |

### Tampilan Home

Tampilan home memiliki fitur dimana user dapat melihat semua berita dan user juga dapat mencari berita yang diinginkan. Tampilan utama dari website tersebut juga memiliki navigation bar yang bertujuan untuk mengkategorikan setiap berita yang ada dan sesuai dengan apa yang terdapat dalam portal berita kompas.com maupun sindonews.com. Saat salah satu pilihan dipilih oleh user, akan ditampilkan tampilan yang sama dengan tampilan home. Akan tetapi, berita yang disajikan hanya berita yang sesuai dengan pilihan kategori user. SQL untuk menampilkan berita dapat dilihat pada Segmen Program 4.24 dan untuk *interface* untuk menampilkan berita dapat dilihat pada Segmen Program 4.25.

Segmen Program 4.24 SQL untuk menampilkan berita home

|  |
| --- |
| <?php  include "database.php";  $limit = 15;  $page = isset($\_GET['page']) ? $\_GET['page'] : 1;  if(empty($page)){  $start = 0;  $page = 1;  }else{$start = ($page - 1) \* $limit;}  $start = ($page - 1) \* $limit;  $sql = "SELECT \* FROM news ORDER BY str\_to\_date(`tanggal`, '%d/%m/%Y') DESC LIMIT $start, $limit";  $res=mysqli\_query($con, $sql);  $news = array();  while ($row = mysqli\_fetch\_assoc($res)) {$news[] = $row;}  $sql = "SELECT count(id) AS id FROM news";  $res = mysqli\_query($con, $sql);  $newsCount = array();  while ($row = mysqli\_fetch\_assoc($res)) {$newsCount[] = $row;}  $total = $newsCount[0]['id'];  $pages = ceil($total/$limit);  $Previous = $page - 1;  $Next = $page + 1; ?> |

Segmen Program 4.25 Home interface

|  |
| --- |
| <?php foreach ($news as $berita): ?>  <div class="row">  <div class="col-md-12">    <a href="detailberita.php?berita=<?php echo $berita['judul'] ?>" style="text-decoration: none; color:black">  <div class="card">  <div class="card-body">  <div class="row">  <div class="col-md-3">  <img src="<?php echo $berita['gambar'] ?>" class="img-thumbnail" style="width:auto; height: 200px">  </div>  <div class="col-md-9">  <p class="judulberita"><?php echo $berita['judul'] ?></p>  <span class="align-bottom"><?php echo $berita['tanggal'] ?></span>  </div>  </div>  </div>  </div>  </a>  </div>  </div>  <?php endforeach ?> |

### Tampilan Detail Berita

Detail berita akan menampilkan detail dari sebuah berita. Detail tersebut mencakup judul berita, tanggal berita, isi berita dan hasil prediksi berita tersebut. SQL untuk menampilkan berita dapat dilihat pada Segmen Program 4.26 dan untuk menampilkan *interface* detail berita dapat dilihat pada Segmen Program 4.27.

Segmen Program 4.26 SQL detail berita

|  |
| --- |
| <?php  include "database.php";  if (isset($\_GET['berita'])){  $judulBerita = $\_GET['berita'];  // print($judulBerita);  $sql = "SELECT \* FROM news WHERE judul='$judulBerita'";  $res = mysqli\_query($con, $sql);  $berita = [];  #while($row =mysqli\_fetch\_assoc($res)){ $judul[] = $row['judul'];$isiberita[] = $row['isi'];}  while ($row = mysqli\_fetch\_assoc($res)) {  $berita = $row;  }  }  ?> |

Segmen Program 4.27 Detail interface

|  |
| --- |
| <div class="row mt-3 mx-3">  <div class="col-md-12">  <div class="row">  <div class="col-md-12">  <!-- Headline -->  <div class="row">  <div class="col-md-12"><p class="judulberita"><?php echo $berita['judul'] ?></p></div>  </div>  <div class="row">  <div class="col-md-12"><p><?php echo $berita['tanggal'] ?></p></div>  </div>  </div>  </div>  </div>  </div>  <div class="row mt-3 mx-3">  <div class="col-md-12">  <p class="isiberita"><?php echo $berita['isi'] ?></p>  </div>  </div>    <div class="row mt-3 mx-3">  <?php $tagBerita = str\_replace("<br>", ",", $berita['prediksi']); ?>  <p>Tags : <span><?php echo $tagBerita; ?></span></p>  </div> |

### Tampilan Search

Tampilan search akan menampilkan berita yang dicari dan *search input* untuk mencari berita yang diinginkan. Fitur ini akan menampilkan berita berdasarkan judul berita. SQL untuk menampilkan berita dapat dilihat pada Segmen Program 4.28 dan untuk menampilkan *interface search* dapat dilihat pada Segmen Program 4.29.

Segmen Program 4.28 SQL search

|  |
| --- |
| <?php  include "database.php";  if (isset($\_POST['search'])){  $beritayangdicari = $\_POST['textBerita'];  }else{  $beritayangdicari = $\_POST['textBerita'];  }  $sql = "SELECT \* FROM news WHERE judul LIKE '%$beritayangdicari%' ORDER BY str\_to\_date(`tanggal`, '%d/%m/%Y') DESC";  $res=mysqli\_query($con, $sql);  $news = array();  while ($row = mysqli\_fetch\_assoc($res)) {  $news[] = $row;  }  ?> |

Segmen Program 4.29 Search interface

|  |
| --- |
| <div class="row mt-5 mx-3">  <div class="col-md-12">  <form autocomplete="off" action="" method="POST" class="d-flex">  <input class="form-control me-2" name="textBerita" type="search" placeholder="Search Berita" aria-label="Search" value="<?php echo $beritayangdicari; ?>">  <!-- <button class="btn btn-outline-success">Search</button> -->  <input type="submit" name="search" class="btn btn-outline-success" value="Search">  </form>  </div>  </div>  <div class="row mt-3 mx-3">  <div class="col-md-12">  <p class="judul">Hasil Pencarian : <?php echo $beritayangdicari ?></p>  <hr style="border:1px solid black">  </div>  </div>  <?php foreach ($news as $berita): ?>  <div class="row mt-3 mx-3">  <div class="col-md-12">  <a href="detailberita.php?berita=<?php echo $berita['judul'] ?>" style="text-decoration: none; color:black">  <div class="card">  <div class="card-body">  <div class="row">  <div class="col-md-3">  <img src="<?php echo $berita['gambar'] ?>" class="img-thumbnail" style="width:auto; height: 200px">  </div>  <div class="col-md-9">  <p class="judulberita"><?php echo $berita['judul'] ?></p>  <span class="align-bottom"><?php echo $berita['tanggal'] ?></span>  </div>  </div>  </div>  </div>  </a>  </div>  </div>  <?php endforeach ?> |

# PENGUJIAN SISTEM

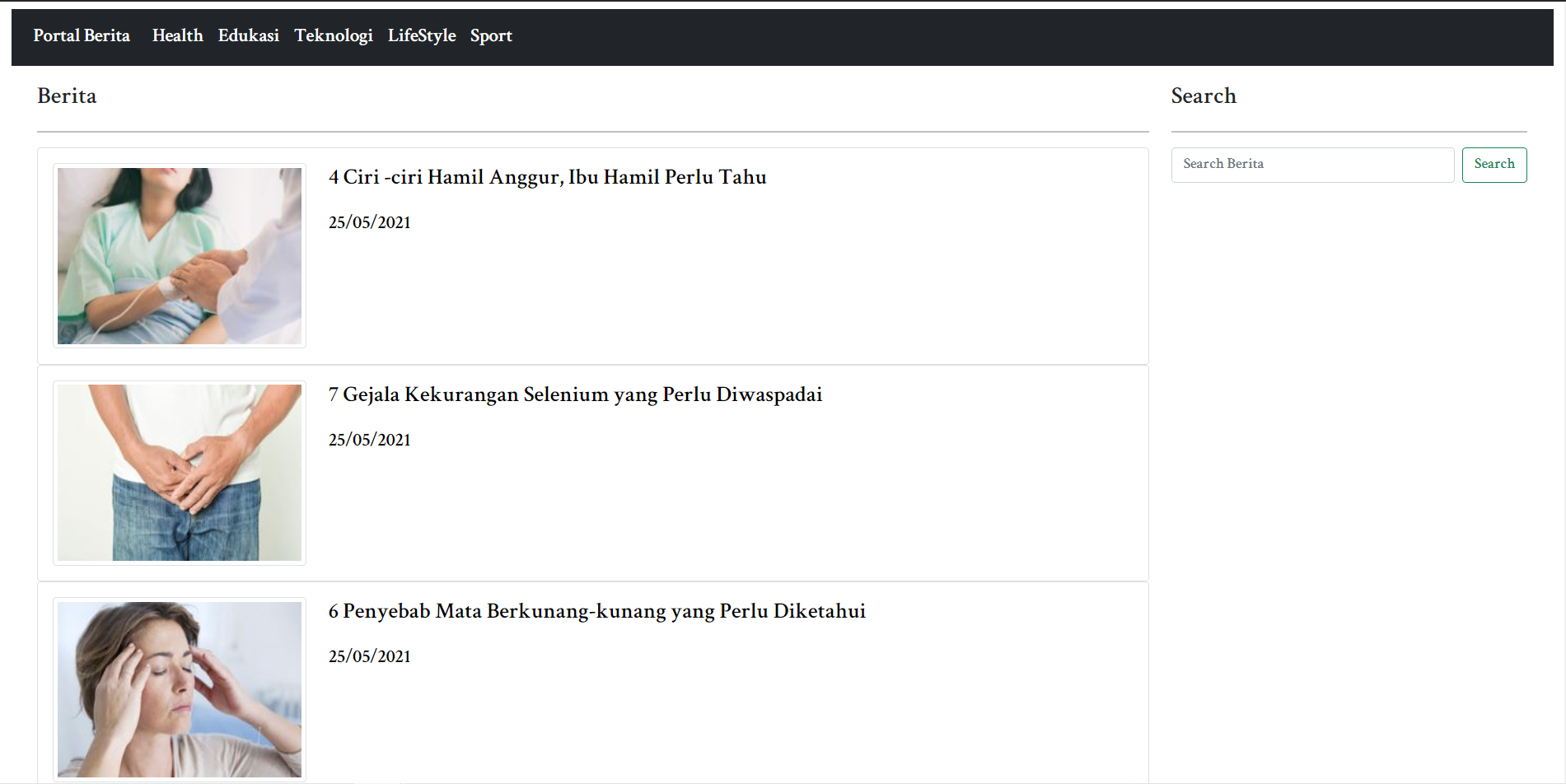
Pada bab ini akan dibahas tentang pengujian dari sistem yang telah dibangun. Pengujian ini terdiri dari pengujian aplikasi, pengujian model dan pengujian matrix. Data yang dipakai didapat dari hasil scraping berita online yaitu kompas.com dan sindonews.com. Data yang dipakai memiliki kategori edukasi, *health, lifestyle, sports*, dan tekno.

## Pengujian Aplikasi

Pengujian ini untuk memastikan bahwa aplikasi dapat berjalan dengan benar. Pengujian ini terdiri dari pengujian *web scraping* untuk mengambil berita dan pengujian klasifikasi untuk memastikan bahwa berita dapat diklasifikasikan dengan baik dan benar.

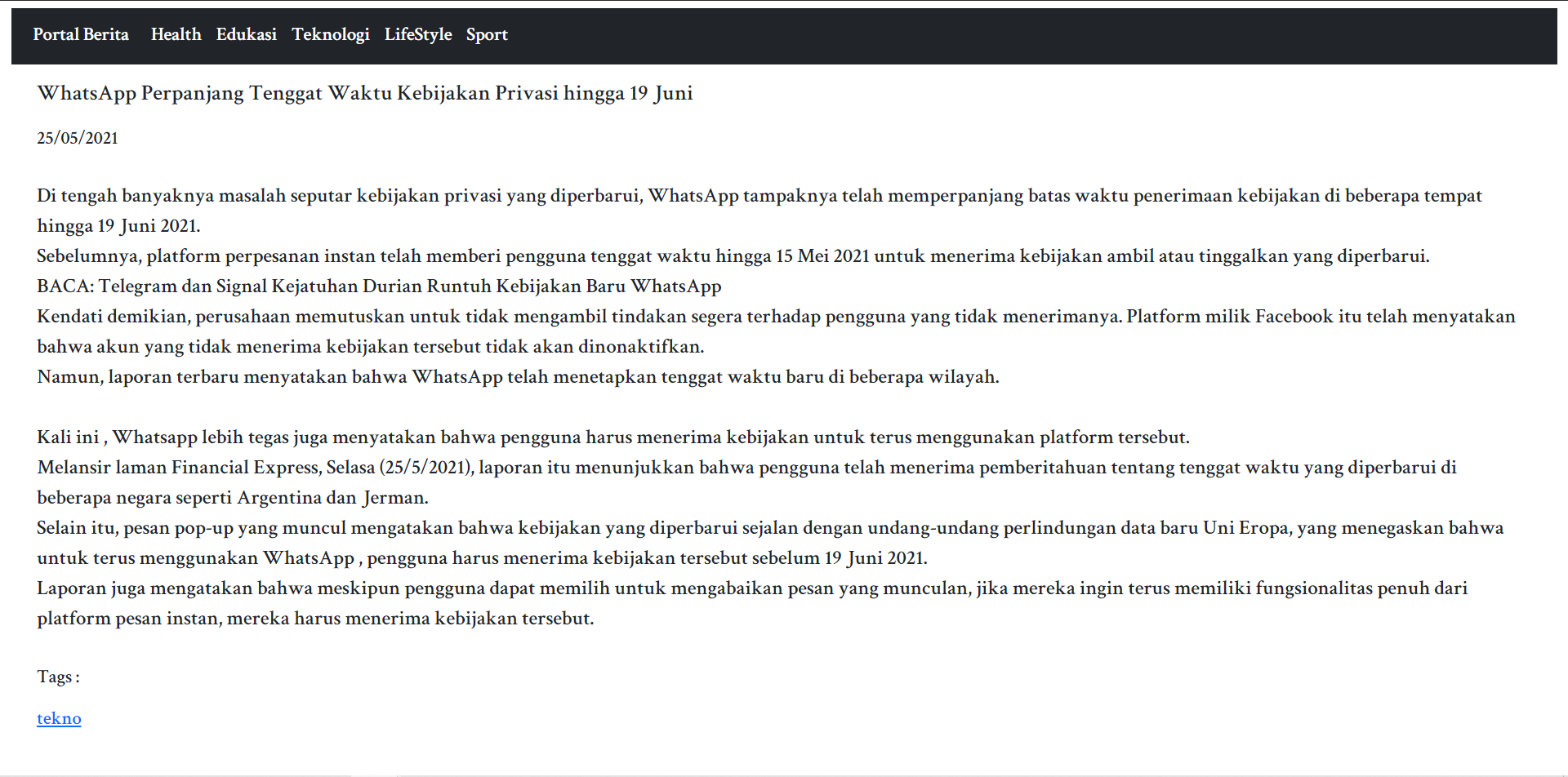
### Pengujian web scraping

Berita yang ambil merupakan berita yang terdapat pada situs kompas.com dan sindonews.com. Setelah berhasil mendapatkan data, data berita yang diambil akan dimasukkan ke dalam database.



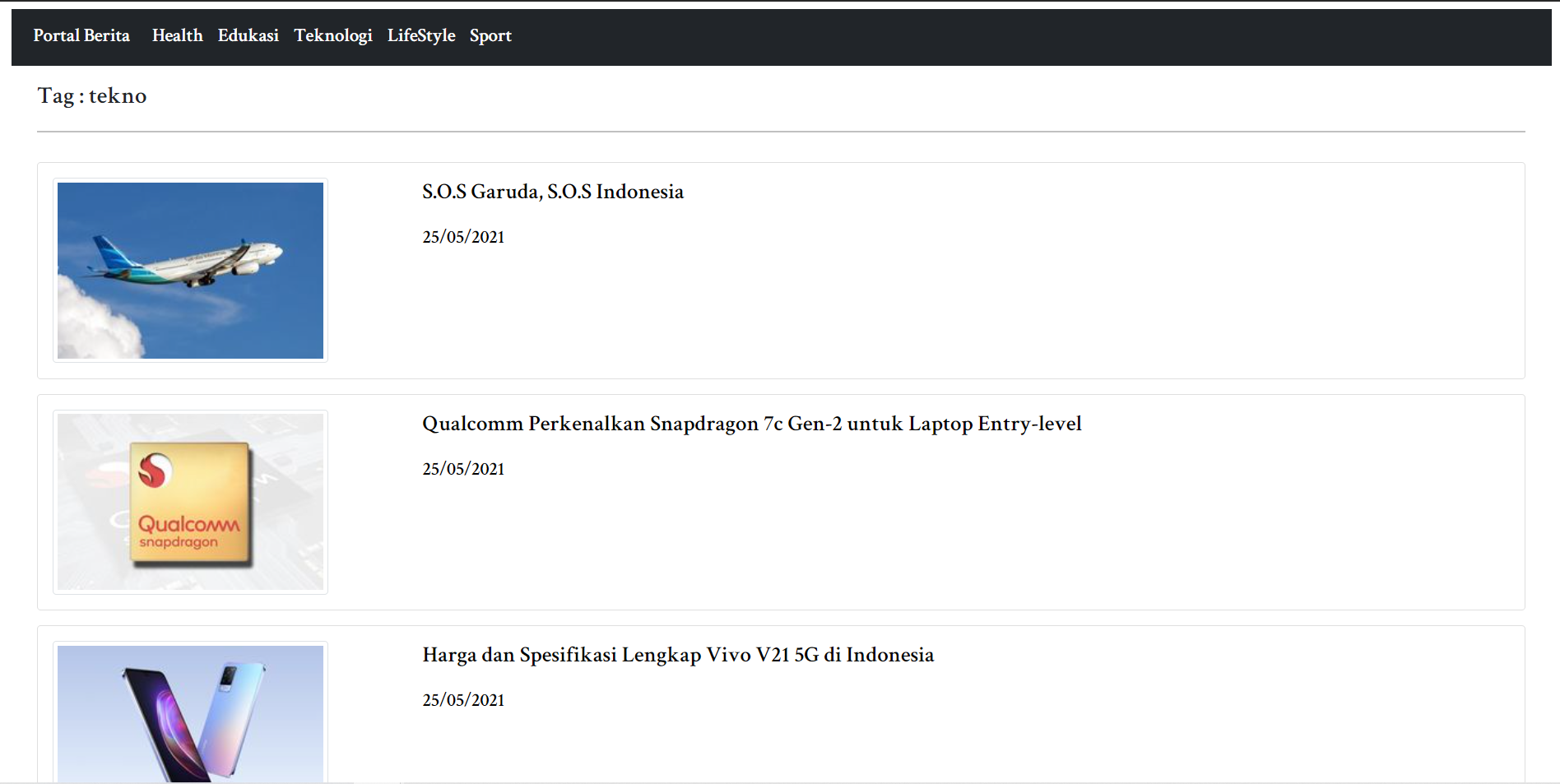
Gambar 5.1 Pengujian scraping berita

Gambar 5.1 merupakan tampilan awal yang berguna untuk menampilkan berita yang ada. Berita yang disajikan merupakan berita yang diambil setiap hari pada saat malam hari. Untuk menampilkan berita setiap hari, *scheduler* akan menjalankan *python script* yang sudah dibuat. *Task* tersebut bertujuan untuk melakukan *scraping* data, melakukan prediksi serta memasukkan ke dalam *database*.



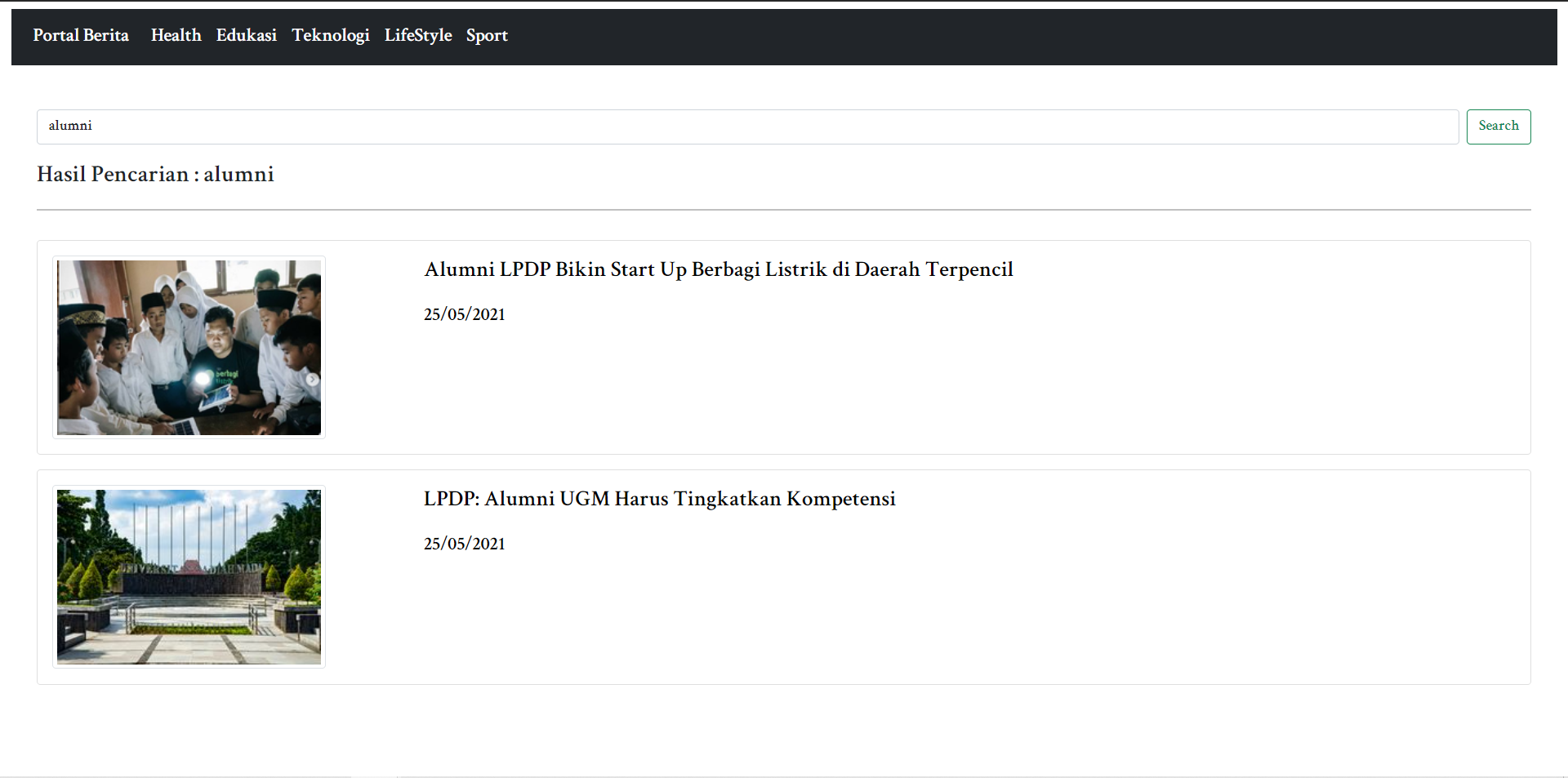
Gambar 5.2 Halaman detail berita

Saat user menekan salah satu berita yang diinginkan, aplikasi akan menampilkan detail dari berita tersebut. Detail tersebut terdiri dari judul berita, tanggal, isi beserta tag berita. *Tag* berita didapatkan dari prediksi yang dihasilkan oleh model yang dibangun. *Tag* dari suatu berita juga dapat ditekan yang bertujuan untuk menampilkan semua berita yang memiliki hasil prediksi yang sama dengan *tag* yang dipilih. Gambar 5.3 akan menampilkan semua berita yang memiliki tag yang sama.



Gambar 5.3 Halaman tag yang dipilih

Aplikasi ini juga memiliki fitur dimana user dapat mencari berita yang diinginkan. Fitur *search* akan mencari berita berdasarkan judul yang diinginkan. Gambar 5.4 merupakan halaman dimana user dapat mencari berita yang diinginkan.



Gambar 5.4 Halaman search

## Pengujian model

Pengujian ini menggunakan data *testing* 20% dari keseluruhan data yang didapat sebelumnya dan menggunakan data baru berjumlah 1285. *Testing* ini berguna untuk mengevaluasi model yang dibangun. Pengujian ini dilakukan untuk mencari konfigurasi terbaik yang didapat dengan melakukan pengujian pada setiap *parameter* yang digunakan. *Hyperparameter* dapat dilihat pada tabel 5.1.

Tabel 5.1 Konfigurasi Hyperparameter

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Bert-base-p1 | | Bert-base-multilingual-uncased |
| Dropout | 0.1-0.5 | | |
| Learning rate | 1e-5, 2e-5, 3e-5, 4e-5 | 2e-5,3e-5,5e-5 | |

Adapun *hyperparameter* yang memiliki nilai konstan yaitu:

* Batch size : 8
* Epoch : 3

### Pengujian pada indobenchmark/bert-base-p1

Pengujian ini menggunakan pretrained model indobenchmark/bert-base-p1 dengan kombinasi *dropout* dan *learning rate*. *Learning rate* yang digunakan merupakan *learning rate* yang digunakan oleh IndoNLU saat *finetuning*. Pengujian ini akan dilihat berdasarkan *training accuracy, testing accuracy* dan akurasi terhadap data baru.

Tabel 5.2 Pengujian pada indobenchmark/bert-base-p1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Learning rate | Dropout | Training Accuracy | Testing Accuracy | Akurasi |
| 1e-5 | 0.1 | 97.206% | 96.196% | 86.303% |
| 1e-5 | 0.2 | 97.384% | 96.988% | 86.770% |
| 1e-5 | 0.3 | 97.642% | 96.909% | 86.147% |
| 1e-5 | 0.4 | 97.127% | 96.354% | 85.836% |
| 1e-5 | 0.5 | 97.582% | 96.038% | 85.992% |
| 2e-5 | 0.1 | 97.305% | 96.196% | 86.303% |
| 2e-5 | 0.2 | 96.790% | 96.117% | 86.926% |
| 2e-5 | 0.3 | 97.642% | 96.513% | 86.381% |
| 2e-5 | 0.4 | 97.681% | 96.038% | 85.369% |
| 2e-5 | 0.5 | 96.849% | 96.195% | 86.692% |
| 3e-5 | 0.1 | 96.651% | 95.483% | 85.914% |
| 3e-5 | 0.2 | 96.710% | 95.721% | 85.214% |
| 3e-5 | 0.3 | 96.829% | 95.800% | 86.147% |
| 3e-5 | 0.4 | 96.909% | 96.117% | 85.525% |
| 3e-5 | 0.5 | 96.750% | 95.483% | 85.447% |
| **4e-5** | **0.1** | **95.957%** | **95.166%** | **87.081%** |
| 4e-5 | 0.2 | 96.235% | 96.434% | 84.747% |
| 4e-5 | 0.3 | 94.234% | 95.562% | 85.525% |
| 4e-5 | 0.4 | 95.106% | 95.245% | 86.692% |
| 4e-5 | 0.5 | 96.136% | 95.245% | 84.980% |

Pada Tabel 5.1 dapat dilihat bahwa *learning rate* dan *dropout* berpengaruh terhadap model yang dibangun. Semakin besar nilai *learning rate* maka semakin cepat model belajar sehingga mendapatkan solusi yang kurang optimal. Hal itu dapat menyebabkan model kurang dapat melakukan prediksi dengan benar dapat dilihat pada Tabel 5.1. Akurasi semakin menurun ketika *learning rate* memiliki nilai yang lebih besar untuk setiap *dropout* yang digunakan. Akurasi tertinggi pada model indobenchmark/indobert-base-p1 adalah 87.081%.

### Pengujian pada bert-base-multilingual-uncased

Pengujian ini menggunakan pretrained model bert-base-multilingual-uncased dengan kombinasi *dropout* dan *learning rate*. *Learning rate* yang digunakan merupakan *learning rate* yang direkomendasikan oleh BERT. Pengujian ini akan dilihat berdasarkan *training loss, testing loss, training accuracy, testing accuracy* dan akurasi terhadap data baru yang didapatkan.

Tabel 5.3 Pengujian pada bert-base-multilingual-uncased

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Learning rate | Dropout | Training Accuracy | Testing Accuracy | Akurasi |
| 2e-5 | 0.1 | 97.008% | 95.404% | 86.147% |
| **2e-5** | **0.2** | **95.759%** | **94.532%** | **87.704%** |
| 2e-5 | 0.3 | 96.374% | 95.404% | 85.525% |
| 2e-5 | 0.4 | 96.037% | 94.294% | 85.525% |
| 2e-5 | 0.5 | 96.275% | 95.245% | 86.225% |
| 3e-5 | 0.1 | 95.680% | 93.502% | 85.758% |
| 3e-5 | 0.2 | 96.453% | 93.977% | 83.501% |
| 3e-5 | 0.3 | 95.442% | 94.611% | 85.680% |
| 3e-5 | 0.4 | 94.888% | 94.215% | 84.591% |
| 3e-5 | 0.5 | 95.343% | 93.343% | 86.770% |
| 5e-5 | 0.1 | 89.875% | 92.393% | 81.945% |
| 5e-5 | 0.2 | 88.369% | 87.876% | 76.186% |
| 5e-5 | 0.3 | 93.619 | 92.947% | 82.723% |
| 5e-5 | 0.4 | 91.638% | 91.759% | 80.700% |
| 5e-5 | 0.5 | 93.837% | 92.789% | 83.891% |

Pada Tabel 5.2 dapat dilihat bahwa *learning rate* dan *dropout* berpengaruh terhadap model yang dibangun. Dari pengujian tersebut, dapat disimpulkan bahwa pretrained model yang dipakai juga berpengaruh terhadap tingkat akurasi. Akurasi tertinggi dengan *pretrained model* bert-base-multilingual-uncased dapat mencapai 87.704% pada learning rate 2e-5 dimana *pretrained model* indobenchmark/indobert-base-p1 dapat mencapai 87.081% pada learning rate 4e-5 (+6.23%). Nilai *dropout* yang teralu kecil / besar dapat mempengaruhi tingkat akurasi dari model yang dibangun. Oleh karena itu, dibutuhkan *dropout* yang sesuai untuk setiap *learning rate* yang dipakai. Akurasi tertinggi pada model bert-base-multilingual-uncased adalah 87.704%.

### Pengujian epoch

Pengujian ini akan menggunakan 3 konfigurasi learning rate dan dropout terbaik pada setiap model yang diambil dari pengujian sebelumnya. Pengujian ini akan menggunakan 2-4 epoch dikarenakan BERT sebelumnya sudah melakukan *pretraining* sehingga yang dibutuhkan hanya *finetuning* untuk melakukan tugas klasifikasi. *Epoch* yang digunakan juga merupakan rekomendasi dari paper. Pengujian ini akan dilihat berdasarkan *training accuracy, testing accuracy* dan akurasi terhadap data baru yang berjumlah 1285.

Tabel 5.4 Pengujian Epoch

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pretrained Model | Learning Rate | Dropout | Epoch | Training Accuracy | Testing Accuracy | Akurasi |
| Indobert-base-p1 | **1e-5** | **0.2** | **2** | **95.601%** | **95.483%** | **88.093%** |
| 1e-5 | 0.2 | 3 | 97.384% | 96.988% | 86.770% |
| 1e-5 | 0.2 | 4 | 98.454% | 96.513% | 85.914% |
| 2e-5 | 0.2 | 2 | 95.541% | 96.354% | 85.369% |
| 2e-5 | 0.2 | 3 | 96.790% | 96.117% | 86.926% |
| 2e-5 | 0.2 | 4 | 98.494% | 95.404% | 85.525% |
| 4e-5 | 0.1 | 2 | 94.491% | 95.721% | 85.680% |
| 4e-5 | 0.1 | 3 | 95.957% | 95.166% | 87.081% |
| 4e-5 | 0.1 | 4 | 96.334% | 94.770% | 85.369% |
| Bert-base-multilinggual-uncased | 2e-5 | 0.2 | 2 | 94.016% | 94.690% | 85.603% |
| 2e-5 | 0.2 | 3 | 95.759% | 94.532% | 87.704% |
| 2e-5 | 0.2 | 4 | 97.364% | 95.087% | 81.712% |
| 2e-5 | 0.5 | 2 | 94.333% | 94.374% | 85.214% |
| 2e-5 | 0.5 | 3 | 96.275% | 95.245% | 86.225% |
| 2e-5 | 0.5 | 4 | 97.562% | 95.166% | 85.214% |
| 3e-5 | 0.5 | 2 | 92.867% | 93.423% | 85.214% |
| 3e-5 | 0.5 | 3 | 95.343% | 93.343% | 86.770% |
| 3e-5 | 0.5 | 4 | 96.671% | 94.294% | 85.136% |

Tabel 5.4 merupakan hasil dari pengujian epoch pada konfigurasi terbaik model sebelumnya. Dari tabel tersebut dapat disimpulkan bahwa semakin besar *epoch* maka semakin baik akurasi yang didapatkan pada saat training data. Akan tetapi, semakin baik pada saat *training*, saat evaluasi model belum tentu mendapatkan hasil yang baik saat diujikan terhadap data baru. Pada pengujian *epoch* hasil terbaik dapat mencapai akurasi 88.093% pada *epoch* 2.

## Pengujian Klasifikasi

Pengujian ini menggunakan data baru berjumlah 1285. Pengujian ini akan menggunakan model indobenchmark/indobert-base-p1 dan dengan konfigurasi terbaik terakhir yang didapatkan saat pengujian model yaitu dengan *learning rate* 1e-5 *dropout* 0.2, dan *epoch* 2.

Tabel 5.5 Metrics pengukuran tiap kategori

|  |  |  |  |
| --- | --- | --- | --- |
| Kategori | Precision | Recall | F1-Score |
| Edukasi | 0.95 | 0.90 | 0.92 |
| Tekno | 0.90 | 0.90 | 0.90 |
| Sports | 0.97 | 0.96 | 0.97 |
| Health | 0.67 | 0.64 | 0.65 |
| Lifestyle | 0.82 | 0.87 | 0.84 |

Tabel 5.5 merupakan metrics pengukuran terhadap data baru. Dari tabel tersebut, dapat disimpulkan bahwa model yang dibangun masih belum bisa melakukan prediksi terhadap kategori *health* dengan baik. Hal ini dikarenakan data yang dipakai pada saat *training* sedikit dan pada situs sindonews.com tidak memiliki kategori *health*.

Tabel 5.6 Confusion Matrix Edukasi

|  |  |  |  |
| --- | --- | --- | --- |
|  | | True Label | |
| Positif | Negatif |
| Predicted label | Positif | 296 | 16 |
| Negatif | 34 | 936 |

Tabel 5.7 Confusion Matrix Tekno

|  |  |  |  |
| --- | --- | --- | --- |
|  | | True Label | |
| Positif | Negatif |
| Predicted label | Positif | 197 | 23 |
| Negatif | 21 | 1044 |

Tabel 5.8 Confusion Matrix Sports

|  |  |  |  |
| --- | --- | --- | --- |
|  | | True Label | |
| Positif | Negatif |
| Predicted label | Positif | 248 | 7 |
| Negatif | 9 | 1021 |

Tabel 5.9 Confusion Matrix Health

|  |  |  |  |
| --- | --- | --- | --- |
|  | | True Label | |
| Positif | Negatif |
| Predicted label | Positif | 68 | 34 |
| Negatif | 39 | 1144 |

Tabel 5.10 Confusion Matrix Lifestyle

|  |  |  |  |
| --- | --- | --- | --- |
|  | | True Label | |
| Positif | Negatif |
| Predicted label | Positif | 323 | 69 |
| Negatif | 50 | 843 |

# KESIMPULAN DAN SARAN

Bab ini berisi tentang kesimpulan dari implementasi *web scraping* dan klasifikasi dalam pembuatan portal berita *online* dengan menggunakan metode *Bidirectional Encoder Representations from Transformers (BERT)*

## Kesimpulan

Dari hasil pembuatan sistem yang dilakukan, dapat ditarik beberapa kesimpulan, antara lain:

* Setiap *website* yang digunakan pada saat melakukan *scraping* memiliki struktur yang berbeda sehingga code yang digunakan untuk mengambil isi dari sebuah *website* yang diinginkan berbeda.
* Kategori yang dipilih lebih baik menggunakan kategori yang terdapat pada semua *website* yang akan digunakan agar tidak kekurangan data tertentu.
* Pretrained model yang digunakan akan mempengaruhi peforma dari model yang dibangun
* Berdasarkan konfigurasi *hyperparameter* yang disarankan / dilakukan oleh IndoNLU (indobert-base-p1) dan BERT (bert-base-multilingual-uncased), dapat disimpulkan bahwa dalam pembuatan model untuk melakukan klasifikasi pada skripsi ini indobert-base-p1 memiliki peforma yang lebih baik daripada bert-base-multilingual-uncased.
* Setiap kombinasi *hyperparameter* yang digunakan dapat mengaruhi kinerja model yang dibangun.
* Akurasi terbaik dalam melakukan klasifikasi dapat mencapai 88.093% dengan konfigurasi *learning rate* 1e-05, *dropout* 0.2, dan *epoch* 2

## Saran

Saran yang dapat diberikan untuk peneliti agar dapat mengembangkan program lebih lanjut, adalah:

* Dataset yang digunakan lebih banyak dan seimbang antar setiap kategori.
* Menambahkan proses *preprocessing* yang dapat meningkatkan kinerja model yang nantinya akan dibangun.
* Dapat menambahkan layer diatas BERT untuk meningkatkan akurasi seperti layer LSTM.

# DAFTAR REFERENSI

Al Sbou, A. M. F. (2018). A survey of Arabic text classification models. *International Journal of Electrical and Computer Engineering*, *8*(6), 4352–4355. https://doi.org/10.11591/ijece.v8i6.pp.4352-4355

Aldwairi, M., & Alwahedi, A. (2018). Detecting fake news in social media networks. *Procedia Computer Science*, *141*, 215–222. https://doi.org/10.1016/j.procs.2018.10.171

Ali Fauzi, M., Arifin, A. Z., Gosaria, S. C., & Prabowo, I. S. (2017). Indonesian news classification using naïve bayes and two-phase feature selection model. *Indonesian Journal of Electrical Engineering and Computer Science*, *8*(3), 610–615. https://doi.org/10.11591/ijeecs.v8.i3.pp610-615

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques. *ArXiv*.

Apuke, O. D., & Omar, B. (2021). Fake news and COVID-19: modelling the predictors of fake news sharing among social media users. *Telematics and Informatics*, *56*(July), 101475. https://doi.org/10.1016/j.tele.2020.101475

Aziz, A., & Rahmah, Y. (2017). Portal system for Indonesian online newspaper - Based feed parser simple pie. *Proceedings - 2016 International Seminar on Application of Technology for Information and Communication, ISEMANTIC 2016*, 169–173. https://doi.org/10.1109/ISEMANTIC.2016.7873832

Boukil, S., Biniz, M., El Adnani, F., Cherrat, L., & Moutaouakkil, A. E. El. (2018). Arabic text classification using deep learning technics. *International Journal of Grid and Distributed Computing*, *11*(9), 103–114. https://doi.org/10.14257/ijgdc.2018.11.9.09

Dogucu, M., & Çetinkaya-Rundel, M. (2020). Web Scraping in the Statistics and Data Science Curriculum: Challenges and Opportunities. *Journal of Statistics Education*, *0*(0), 1–24. https://doi.org/10.1080/10691898.2020.1787116

Fang, W., Luo, H., Xu, S., Love, P. E. D., Lu, Z., & Ye, C. (2020). Automated text classification of near-misses from safety reports: An improved deep learning approach. *Advanced Engineering Informatics*, *44*(March 2019), 101060. https://doi.org/10.1016/j.aei.2020.101060

Flores, V. A., Permatasari, P. A., & Jasa, L. (2020). Penerapan Web Scraping Sebagai Media Pencarian dan Menyimpan Artikel Ilmiah Secara Otomatis Berdasarkan Keyword. *Majalah Ilmiah Teknologi Elektro*, *19*(2), 157. https://doi.org/10.24843/mite.2020.v19i02.p06

HaCohen-Kerner, Y., Miller, D., & Yigal, Y. (2020). The influence of preprocessing on text classification using a bag-of-words representation. *PLoS ONE*, *15*(5), 1–22. https://doi.org/10.1371/journal.pone.0232525

Hartmann, J., Huppertz, J., Schamp, C., & Heitmann, M. (2019). Comparing automated text classification methods. *International Journal of Research in Marketing*, *36*(1), 20–38. https://doi.org/10.1016/j.ijresmar.2018.09.009

Kannan, S., Gurusamy, V., Vijayarani, S., Ilamathi, J., Nithya, M., Kannan, S., & Gurusamy, V. (2015). Preprocessing Techniques for Text Mining. *International Journal of Computer Science & Communication Networks*, *5*(1), 7–16.

Karthikeyan, T., Sekaran, K., Ranjith, D., Vinoth kumar, V., & Balajee, J. M. (2019). Personalized content extraction and text classification using effective web scraping techniques. *International Journal of Web Portals*, *11*(2), 41–52. https://doi.org/10.4018/IJWP.2019070103

Kasanah, A. N., Muladi, M., & Pujianto, U. (2019). Penerapan Teknik SMOTE untuk Mengatasi Imbalance Class dalam Klasifikasi Objektivitas Berita Online Menggunakan Algoritma KNN. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *3*(2), 196–201. https://doi.org/10.29207/resti.v3i2.945

Kwak, K. T., Hong, S. C., & Lee, S. W. (2020). A study of repetitive news display and news consumption in Korea. *Telematics and Informatics*, *46*(October 2019), 101313. https://doi.org/10.1016/j.tele.2019.101313

Mulahuwaish, A., Gyorick, K., Ghafoor, K. Z., Maghdid, H. S., & Rawat, D. B. (2020). Efficient classification model of web news documents using machine learning algorithms for accurate information. *Computers and Security*, *98*. https://doi.org/10.1016/j.cose.2020.102006

Ouatik, S., Alaoui, E., & Nahnahi, N. E. (2021). Contextual Semantic Embeddings based on Fine-tuned AraBERT Model for Arabic Text Multi-class Categorization. *Journal of King Saud University - Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2021.02.005

Paul, S., & Saha, S. (2020). CyberBERT: BERT for cyberbullying identification: BERT for cyberbullying identification. *Multimedia Systems*, *0123456789*. https://doi.org/10.1007/s00530-020-00710-4

Peng, Y., Yan, S., & Lu, Z. (2019). Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. *ArXiv*, *iv*. https://doi.org/10.18653/v1/w19-5006

Pramudita, Y. D., Putro, S. S., & Makhmud, N. (2018). Klasifikasi Berita Olahraga Menggunakan Metode Naïve Bayes dengan Enhanced Confix Stripping Stemmer. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(3), 269. https://doi.org/10.25126/jtiik.201853810

Priyanto, A., & Ma’arif, M. R. (2018). Implementasi Web Scrapping dan Text Mining untuk Akuisisi dan Kategorisasi Informasi dari Internet (Studi Kasus: Tutorial Hidroponik). *Indonesian Journal of Information Systems*, *1*(1), 25–33. https://doi.org/10.24002/ijis.v1i1.1664

Sari, W. K., Rini, D. P., & Malik, R. F. (2020). Text Classification Using Long Short-Term Memory With GloVe Features. *Jurnal Ilmiah Teknik Elektro Komputer Dan Informatika*, *5*(2), 85. https://doi.org/10.26555/jiteki.v5i2.15021

Sari, W. K., Rini, D. P., Malik, R. F., & Azhar, I. S. B. (2017). *Klasifikasi Teks Multilabel pada Artikel Berita Menggunakan Long Short- Term Memory dengan Word2Vec*. *1*(10), 276–285.

Sayed, M., Salem, R. K., & Khder, A. E. (2019). A survey of Arabic text classification approaches. *International Journal of Computer Applications in Technology*, *59*(3), 236–251. https://doi.org/10.1504/IJCAT.2019.098601

Setiawan, A., Santoso, L. W., & Adipranata, R. (2020). Klasifikasi Artikel Berita Bahasa Indonesia Dengan Naive Bayes Classifier. *Jurnal Infra*, *8*(1), 146–151.

Sistem, R. (2021). Model Text-Preprocessing Komentar Youtube Dalam Bahasa Indonesia. *JURNAL RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *1*(10), 648–654.

Utomo, F. S., Suryana, N., & Azmi, M. S. (2020). Stemming impact analysis on Indonesian Quran translation and their exegesis classification for ontology instances. *IIUM Engineering Journal*, *21*(1), 33–50. https://doi.org/10.31436/iiumej.v21i1.1170

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017*-*Decem*(Nips), 5999–6009.

Westergaard, D., Stærfeldt, H. H., Tønsberg, C., Jensen, L. J., & Brunak, S. (2018). A comprehensive and quantitative comparison of text-mining in 15 million full-text articles versus their corresponding abstracts. *PLoS Computational Biology*, *14*(2), 1–16. https://doi.org/10.1371/journal.pcbi.1005962

Wilie, B., Vincentio, K., Winata, G. I., Cahyawijaya, S., Li, X., Lim, Z. Y., Soleman, S., Mahendra, R., Fung, P., Bahar, S., & Purwarianti, A. (2020). IndoNLU: Benchmark and resources for evaluating indonesian natural language understanding. *ArXiv*.