Panduan Pelaksanaan Pertemuan Project DTS: Deep Learning

Pendahuluan

Project merupakan sebuah praktik mandiri yang diberikan ke peserta. Pelaksanaan project ditujukan untuk menilai seberapa jauh pemahaman yang dimiliki oleh peserta mengenai teori yang telah didapatkannya. Sebuah studi kasus akan diberikan kepada peserta dan kemampuan analisis serta pemecahan masalah peserta akan diuji secara langsung.

Terdapat dua buah pelaksanaan project yang harus dilakukan peserta, keduanya berlangsung di hari sebelum ujian (pertengahan dan akhir). Project pertama akan disebut sebagai Project Machine Learning, sedangkan project kedua akan disebut dengan Project Deep Learning.

Peraturan

Berikut merupakan beberapa peraturan dalam pelaksanaan project:

1. Pelaksanaan Project:
    a. Project Machine Learning dilakukan secara individu dan dilaksanakan 1 hari.
    b. Project Deep Learning dilakukan secara kelompok dan dilaksanakan selama 2 hari. Selain berupa studi kasus, presentasi dilakukan kelompok di hari ke-2.
2. File yang dikumpulkan:
    a. Project Machine Learning dikerjakan menggunakan Jupyter Notebook (Google Colab) dan disimpan kedalam file dengan format .ipynb. Project dikumpulkan ke Instruktur/Asisten.
    b. Khusus untuk Project Deep Learning, selain file berformat .ipymb, file PowerPoint dengan format .ppt/.pptx juga dikumpulkan ke Instruktur/Asisten.
3. Project dilaksanakan secara mandiri tanpa arahan dari Instruktur/Asisten. Seluruh informasi permasalahan yang harus diselesaikan terjabarkan dalam studi kasus.
4. Peserta diperbolehkan mencari informasi dokumentasi library yang akan ia pakai menggunakan koneksi Internet.
5. Khusus untuk Project Machine Learning, peserta tidak boleh mencari penyelesaian masalah yang tersedia secara online. Instruktur/Asisten harus mengawasi pekerjaan yang dilakukan oleh peserta.

Persiapan

Terdapat beberapa hal yang perlu disiapkan untuk keberlangsungan pelaksanaan project, antara lain:

1. Koneksi internet dan akses menuju Google Colabs
   Koneksi internet diwajibkan tersedia agar peserta dapat mengerjakan projectnya dengan Google Colabs. Koneksi internet juga memungkinkan peserta untuk mencari informasi dokumentasi library yang ia pakai.

2. Project Description
   Project Description merupakan dokumen yang berisi deskripsi dari project yang harus diselesaikan oleh peserta. Terdapat beberapa studi kasus, permasalahan, dan informasi data dan algoritma yang akan digunakan ada didalamnya. Project Description terlampir dan boleh di bagikan untuk para peserta, boleh softcopy maupun hardcopy.

3. Dataset
   Setiap studi kasus akan menggunakan dataset yang berbeda-beda. Ada beberapa dataset yang berukuran kecil ada juga dataset yang berukuran sangat besar (khususnya dataset untuk menunjang Deep Learning). Dataset berukuran besar harus sudah dipersiapkan oleh panitia dan didistribusikan secara offline paling lambat 1 hari sebelum hari pelaksanaan project.

Penilaian

Berikut merupakan tabel penilaian project. Instruktur/Asisten diwajibkan untuk membuka file .ipymb yang dikerjakan oleh peserta dan memeriksa apakah poin-poin yang disebutkan dalam tabel berikut dibuat oleh peserta.

Tabel 1 Tabel Penilaian

Pokok Penilaian		Deskripsi	Nilai
Pre-processing	Melihat data	Setelah loading dataset, peserta menampilkan isi data (print standar, maupun menggunakan pandas)	2
	Memeriksa data	Peserta melakukan pemeriksaan jika terdapat record / field yang tidak layak dipakai, seperti record / field yang bernilai NaN atau field yang berbentuk kategori (yang tidak dapat diproses oleh classifier model)	2
	Visualisasi data	Peserta mampu menampilkan visualisasi sederhana seperti plot (melihat korelasi antar 2 var), barchart (jumlah data dalam variable tertentu), dll.	1

Training	Memecah data	Peserta memecah data menjadi x_train, y_train, x_test, y_test, menggunakan scikit learn.	2
	Train (Fitting)	Peserta mampu menggunakan classifier (dan parameternya) yang sesuai dengan yang diperintahkan dalam *project description*.	3
	Prediksi	Peserta memanfaatkan model yang telah ia *train* untuk melakukan prediksi menggunakan x_test.	2
Evaluating	Confusion Matrix	Peserta dapat menggunakan confusion matrix sebagai alat untuk melakukan evaluasi modelnya, apakah telah di-train dengan baik atau tidak.	2
	Accuracy / Precision / Recall / F1	Peserta dapat menggunakan beberapa metric yang tersedia untuk mengevaluasi apakah modelnya telah di-train dengan baik atau tidak.	2
	Visualisasi Hasil	Peserta mampu melakukan visualisasi performa dari model yang dibuat	1
Dokumentasi		Keterangan yang ditulis oleh peserta untuk menjelaskan output-output dari snippet-code yang dibuatnya. 0 = Tidak ada 1 = Sangat minim 2 = Cukup Lengkap 3 = Sangat Lengkap	3
Presentasi		Performa presentasi yang dilakukan dan pemahaman yang dimiliki oleh kelompok peserta 1 = minim 2 = kurang 3 = cukup 4 = baik 5 = sangat baik	5
Total			25

Lampiran 1 Project Description: DL

Klasifikasi Pakaian dan Aksesoris

Klasifikasi gambar digunakan dalam beberapa aplikasi, mulai dari mengenali penyakin yang mengancam jiwa dalam pemindaian medis, hingga mendeteksi

Dataset MNIST (bisa dibilang) adalah dataset yang paling sering digunakan sebagai permulaan pembelajaran klasifikasi gambar. Komunitas data saintis menyukai dataset ini dan menggunakannya sebagai tolak ukur untuk memvalidasi algoritma mereka. Faktanya, MNIST seringkali menjadi dataset pertama para peneliti yang dicoba.

> *"Jika algoritmanya tidak bisa bekerja pada MNIST, algoritma tersebut pasti tidak akan bisa bekerja untuk dataset lainnya. Jika algoritmanya bekerja pada MNIST, permulaan yang bagus, namun bukan berarti dapat bekerja untuk dataset lainnya."*

Meningkatnya popularitas MNIST selama beberapa tahun terakhir memunculkan beberapa masalah yang menyebabkan MNIST sepertinya akan segera tergantikan:

1. MNIST terlalu mudah! Convolutional nets dapat mencapai akurasi 99.7% pada MNIST, algoritma klasik machine learning juga dapat mencapai akurasi 97%

2. MNIST terlalu sering digunakan. Hampir setiap orang yang memiliki pengalaman Deep Learning pasti pernah memakai dataset MNIST setidaknya sekali. Peneliti Google Brain Research dan pakar deep learning Ian Goodfellow mengatakan bahwa semua orang harus "move-on" dari MNIST

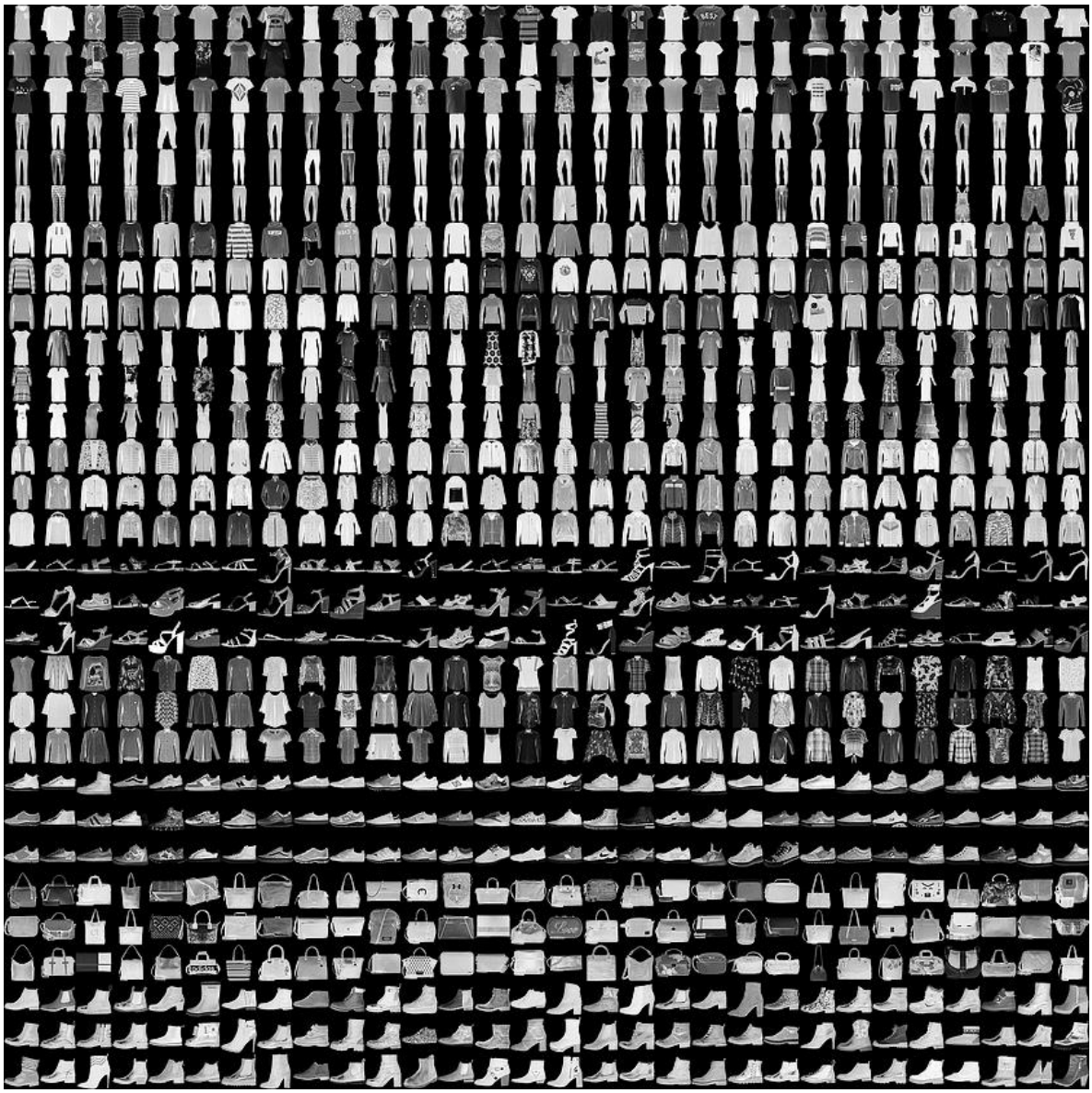
3. MNIST tidak merepresentasikan permasalahan Computer Vision modern, seperti yang dikatakan oleh pemilik library KERAS, François Chollet.

Peneliti dari Zalando (sebuah perusahaan e-commerce) telah mengembangkan dataset klasifikasi gambar baru yang disebut dengan Fashion MNIST dengan harapan dapat menggantikan MNIST. Dataset baru ini berisi gambar baju dan aksesoris, seperti: kemeja, tas, sepatu, dan barang mode lainnya.

Fashion MNIST berisi 55,000 training set dan 10,000 test set. Serupa dengan MNIST, setiap data Fashion MNIST merupakan sebuah gambar skala abu (grayscale) berukuran 28x28 pixel. Masing-masing data memiliki label asosiasinya yang terdiri dari 10 kelas

Tabel 2 Label dan Deskripsi FMNIST

Label	Deskripsi
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle Boot



Gambar 1 Sample Data FMNIST

What to do

1. Bentuk kelompok (banyaknya anggota per kelompok ditentukan oleh instruktur), dan diskusikan kasus ini bersama anggota kelompok kalian.
2. Berdasarkan data-data yang tersedia dalam dataset, lakukan klasifikasi terhadap 10 kelas yang tertera dalam Tabel 1. Dari antara model-model Deep Learning yang telah kalian pelajari, pilih salah satu yang paling cocok digunakan untuk menyelesaikan kasus ini.
3. Laporkan pekerjaan kalian dalam bentuk presentasi (ke dalam file .ppt/.pptx) dan bersainglah dengan kelompok lainnya untuk menemukan model dan arsitektur yang dapat memberikan nilai akurasi yang tinggi!

Catatan Teknis (Cara Mendapatkan Dataset)

Berikut merupakan spesifikasi file dari dataset FMNIST, data tersebut dipublikasikan secara terbuka dan online

Nama File	Isi	# Sample	Ukuran
`train-images-idx3ubyte.gz`	training set images	60,000	26 MBytes
`train-labels-idx1-byte.gz`	training set labels	60,000	29 KBytes
`t10k-images-idx3-ubyte.gz`	test set images	10,000	4.3 MBytes
`t10k-labels-idx1-ubyte.gz`	test set labels	10,000	5.1 KBytes

Untuk mendapatkan data-data tersebut gunakan beberapa snippet code berikut.

```
1  !mkdir data/fashion
2  !wget -O data/fashion/train-images-idx3-ubyte.gz http://fashion-mnist.s3-
   website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz
3  !wget -O data/fashion/train-labels-idx1-ubyte.gz http://fashion-mnist.s3-
   website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz
4  !wget -O data/fashion/t10k-images-idx3-ubyte.gz http://fashion-mnist.s3-
   website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz
5  !wget -O data/fashion/t10k-labels-idx1-ubyte.gz http://fashion-mnist.s3-
   website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz
```

Untuk melakukan import data-data tersebut kedalam sebuah variable gunakan snippet code berikut

```
1  from tensorflow.examples.tutorials.mnist import input_data
2  fashion_mnist = input_data.read_data_sets('data/fashion')
```

Berikut merupakan detail dari data yang telah diimpor

```
1  type(fashion_mnist)
```

Output: `tensorflow.contrib.learn.python.learn.datasets.base.Datasets`

1. Data Training, (**`fashion_mnist.train`**), data yang digunakan untuk melatih model untuk mencari pola masing-masing data berdasarkan class-nya.
   - ✓ 55,000 sampel data
   - ✓ **`fashion_mnist.train.images`** untuk input
   - ✓ **`fashion_mnist.train.labels`** untuk output

2. Data Validation, (**`fashion_mnist.validation`**), umumnya digunakan untuk mencegah model menjadi overfitting, opsional untuk dipakai.
   - ✓ 5,000 sampel data
   - ✓ **`fashion_mnist.validation.images`** untuk input
   - ✓ **`fashion_mnist.validation.labels`** untuk output

3. Data Testing, (**`fashion_mnist.test`**), digunakan untuk mengevaluasi performa dan akurasi dari model, mencerminkan situasi nyata.
   - ✓ 10,000 sampel data
   - ✓ **`fashion_mnist.test.images`** untuk input
   - ✓ **`fashion_mnist.test.labels`** untuk output

Lampiran 2 Project Description: DL

Language Modelling

Language Modelling ada sebuah pekerjaan yang ditujukan untuk memodelkan sebuah bahasa. Language modelling dapat dikelompokkan menjadi dua buah:

1. Character Level Language Modelling, memodelkan bahasa dalam level karakter.
2. Word Level Language Modelling, memodelkan bahasa dalam level kata.

Perbedaan mendasar dari kedua kelompok language modelling tersebut adalah struktur atomik dari bahasanya. Character level mengasumsikan karakter sebagai atom dari kalimat, sedangkan Word level mengasumsikan kata sebagai atom dari kalimat.

Karena Character Level Language Modelling telah didemonstrasikan saat praktikum RNN/LSTM, dalam project ini kalian akan membuat *Word Level Language Modelling*. Inti dari language modelling adalah melakukan prediksi kata selanjutnya berdasarkan kata sebelumnya dan konteks kalimat.

Sebagai contoh :

"Hobi saya makan, saya sangat tidak senang kalau saya ..... "

Word Level Language Modelling harus dapat melakukan prediksi kata selanjutnya dari kalimat diatas. Melihat dari konteks, maka kata setelah kata "saya" diakhir kalimat tersebut seharusnya diisi dengan kata "lapar"

What to do

1. Bentuk kelompok (banyaknya anggota per kelompok ditentukan oleh instruktur), dan diskusikan kasus ini bersama anggota kelompok kalian.
2. Buat model LSTM yang mampu melakukan prediksi kalimat berdasarkan dataset yang telah diberikut. Sebagai contoh, hasil yang harus didapatkan adalah seperti berikut:

   Input: `on a warm autumn day because the stock market was so quiet`
   Prediksi: `on a warm autumn day since it was market which rising a fund`

3. Laporkan pekerjaan kalian dalam bentuk presentasi (ke dalam file .ppt/.pptx) dan bersainglah dengan kelompok lainnya untuk menemukan model dan arsitektur yang dapat memberikan nilai akurasi yang tinggi!

Catatan Teknis (Cara Mendapatkan Dataset)

1. Class Reader

Dataset Pentree Bank merupakan sebuah kumpulan teks yang sangat panjang, untuk melihatnya, kalian bisa akses link berikut

https://raw.githubusercontent.com/tomsercu/lstm/master/data/ptb.train.txt

Kita membutuhkan sebuah class yang ditujukan untuk mengelola teks-teks tersebut sehingga memudahkan kita untuk mengoperasikan data tersebut. Berikut merupakan snippet code untuk mendapatkan class tersebut.

Listing 1 Download dan Unzip Class Reader

```
!mkdir -p /resources/data
!wget -q -O /resources/data/ptb.zip
https://ibm.box.com/shared/static/z2yvmhbskc45xd2a9a4kkn6hg4g4kj5r.zip
!unzip -o /resources/data/ptb.zip -d /resources/
!cp /resources/ptb/reader.py .
```

Class yang akan membantu kalian untuk mengelola dataset Pentree Bank bernama **`reader`**. Untuk menggunakan class tersebut, kalian harus import terlebih dahulu.

Listing 2 Import Class Reder

```
import reader
```

2. Download Dataset

Sekarang saatnya kita melakukan download dataset Pentree Bank, dataset ini yang akan kita gunakan dalam project ini. Gunakan snippet code berikut:

Listing 3 Download Dataset

```
!wget http://www.fit.vutbr.cz/~imikolov/rnnlm/simple-examples.tgz
!tar xzf simple-examples.tgz -C /resources/data/
```

3. Membaca Dataset

Untuk membaca dataset agar dapat digunakan dalam program, gunakan code berikut:

Listing 4 Cara Membaca Dataset dan Memisahkannya

```
# Direktori menuju dataset yang barusan kita download
data_dir = "/resources/data/simple-examples/data/"
# Mengambil data
raw_data = reader.ptb_raw_data(data_dir)
# memisahkan data menjadi 5 buah tipe
train_data, valid_data, test_data, vocabulary, word_to_id = raw_data
```

Berikut merupakan deskripsi detail dari dataset kita

1. train_data : list kata dalam bentuk integer (id) untuk kepentingan training
2. valid_data : list kata dalam bentuk integer (id) untuk kepentingan validasi
3. test_data : list kata dalam bentuk integer (id) untuk kepentingan testing
4. vocabulary : total vocabulary (kata) yang kita miliki
5. word_to_id : dictionary yang memetakan kata ke id

Nama variable	*Class*	*Length*	*Value*
Train_data	List	929589	[9970, 9971, 9972, 9974 ...]
Valid_data	List	73760	[102, 14, 24, 32, ...]
Test_data	List	82430	[1132, 93, 358, 5, ...]
Vocabulary	Int	1	10000
Word_to_id	Dict	10000	{'yellow': 6274, 'del.': 6835, 'four': 346, 'woods': 7818, 'mirage': 5734, ... }

4. Membuat id_to_word

Untuk keperluan tertentu, kita akan membutuhkan dictionary **id_to_word**, yang akan memetakan id ke kata. Sebagai contoh, untuk menampilkan hasil kita nanti. Berikut merupakan snipped code untuk membuat **id_to_word** berdasarkan **word_to_id**.

```
1  # key dan value dalam dictionary dibalik
2  id_to_word = dict(zip(word_to_id.values(), word_to_id.keys()))
```

Seperti yang kita tahu, train_data berisi bentuk integer dari kata, untuk menampilkannya dalam bentuk kata, kita bisa gunakan id_to_word. Perhatikan snippet code berikut.

```
1  # lihat kata ke 100 sampai 110
2  print(" ".join([id_to_word[x] for x in train_data[100:110]]))
```

5. Mengambil Data dalam Batch

Jika arsitektur LSTM memiliki jumlah time-step sebesar 20, maka dalam satu kali feed-forward, LSTM akan membutuhkan 20 kata.

Jika arsitektur LSTM ditetapkan memiliki batch-size sebesar 30, maka dalam satu kali feed-forward, LSTM akan membutuhkan [30 x 20] kata. Perhatikan Gambar 2 untuk ilustrasi yang lebih jelas.

Untuk mempersiapkan input sesuai dengan arsitektur LSTM yang akan kalian buat (sebagai contoh, dengan time-step = 20, dan batch-size = 30), gunakan snipped code berikut:

Listing 5 Mempersiapkan Input Sesuai Arsitektur LSTM yang Ditetapkan

```
1  # lihat kata ke 100 sampai 110
2  iterator = reader.ptb_iterator(train_data, batch_size, num_steps)
3  inputs, targets = iterator.next()
```

Dimana:

- **`inputs`** merupakan kumpulan numpy array kata dengan shape: `(batch-size, time-step)` yang akan di feed ke dalam LSTM kalian.

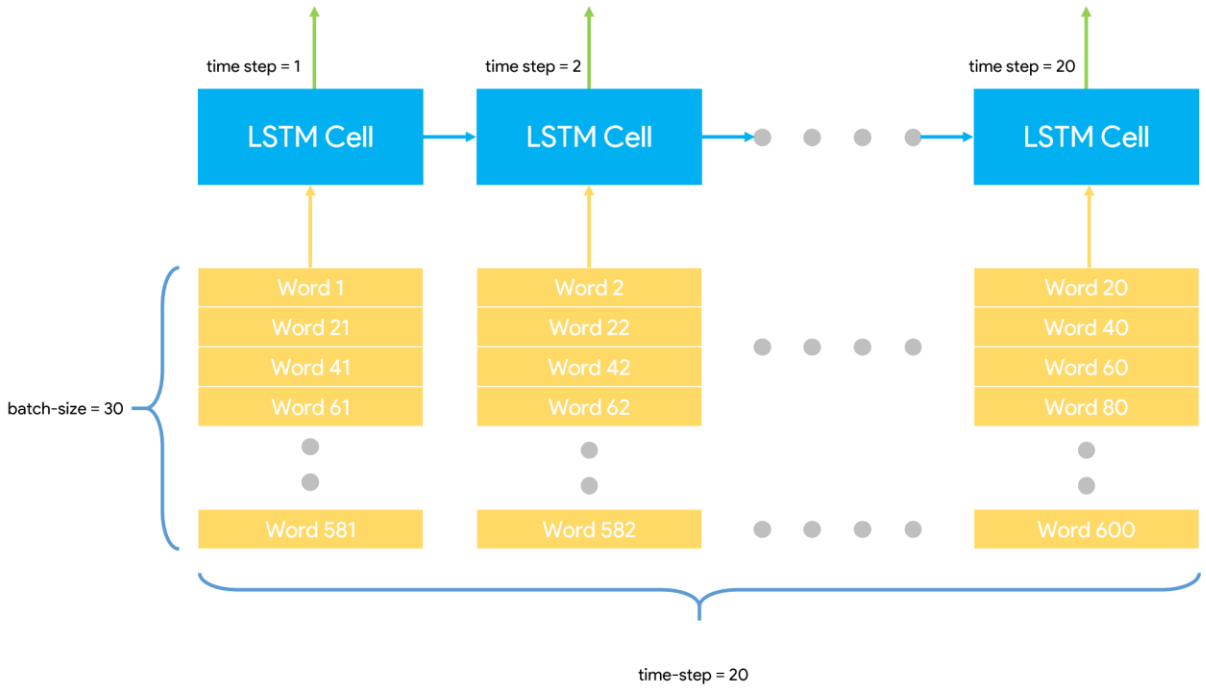
  Sebagai contoh:

```
[ [ word001, word002, word003, ..., word020],
  [ word021, word022, word023, ..., word040],
  ...
  [ word581, word582, word583, ..., word600] ]
```

- **`targets`** sama dengan inputs, namun dengan dengan time-shift sebesar 1. Targets akan digunakan untuk mengevaluasi hasil prediksi berdasarkan kata input.

  Sebagai contoh:

```
[ [ word002, word003, word004, ..., word021],
  [ word022, word023, word024, ..., word041],
  ...
  [ word582, word583, word584, ..., word601] ]
```



Gambar 2 Illustrasi Input untuk LSTM