

Aplikasi Pencarian Dokumen Lintas Bahasa (Indonesia-Inggris)

Alat Bantu Penelitian Untuk Bidang Cross-Language Information Retrieval (CLIR) Dengan Menggunakan Query Expansion (QE)

python

Main tab

☒ Sistem QE

Cross-Language Information Retrieval

Masukkan kueri disini

Kueri awal: -



Hasil kueri ekspansi teratas;

1. -
2. -
3. -

☐ Semua kueri
☐ Kueri awal
☐ Kueri ekspansi 1
☐ Kueri ekspansi 2
☐ Kueri ekspansi 3

Hasil Pencarian dengan Kueri - : x artikel

No Dokumen	Artikel	Cos Similarity
No Dokumen		

python

Main tab

☒ Sistem QE

Cross-Language Information Retrieval

badai salju

Kueri awal: snowstorm



Hasil kueri ekspansi teratas;

1. blizzard
2. -
3. -

☐ Semua kueri
☐ Kueri awal
☐ Kueri ekspansi 1
☐ Kueri ekspansi 2
☐ Kueri ekspansi 3

Hasil Pencarian dengan gabungan semua kueri: 7 artikel

No Dokumen	Artikel	Cos Similarity
1	937 (CNN) New York dan Boston akan menutup sekolah umum mereka pada ...	1.00
2	966 (CNN) Timur Laut dihantam Kamis dengan apa yang tampaknya menjadi ...	1.00
3	8528 (CNN) Ini bukan badai salju biasa, kata pakar cuaca. Layanan Cuaca Nasion...	1.00
4	8549 (CNN) Badai salju besar pertama tahun 2015 telah melanda Amerika Serika...	1.00
5	1730 (CNN) Dengan badai salju yang meluncur ke arah Timur Laut, wilayah ...	1.00
6	1747 (CNN) Sebuah nor'easter sebagian besar terhindar dari New York City dan ...	1.00

python

Main tab

☒ Sistem QE

Cross-Language Information Retrieval

badai salju

Kueri awal: snowstorm



Hasil kueri ekspansi teratas;

1. blizzard
2. -
3. -

☐ Semua kueri
☐ Kueri awal
☒ Kueri ekspansi 1
☐ Kueri ekspansi 2
☐ Kueri ekspansi 3

Hasil Pencarian dengan Kueri 'blizzard': 4 artikel

No Dokumen	Artikel	Cos Similarity
1	8549 (CNN) Badai salju besar pertama tahun 2015 telah melanda Amerika Serika...	1.00
2	1730 (CNN) Dengan badai salju yang meluncur ke arah Timur Laut, wilayah ...	1.00
3	1747 (CNN) Sebuah nor'easter sebagian besar terhindar dari New York City dan ...	1.00
4	8565 (CNN) Sersan. Jennifer Bruno tahu badai salju yang melanda New England ...	1.00



python

Main tab

Dok 1730

Kawat gigi timur laut untuk badai salju

(CNN) Dengan badai salju yang meluncur ke arah Timur Laut, wilayah tersebut meningkatkan persiapan dan mempercepat peringatan Senin malam, menutup sekolah, membatalkan penerbangan, dan menyuruh penduduk bersiap-siap. "Ini adalah badai yang akan datang," kata Walikota Boston Marty Walsh, Senin, pada konferensi pers tentang persiapan badai. "Kita akan dipukul." Di antara perkembangan utama: — Distrik sekolah Boston, Philadelphia, dan New York City akan ditutup Selasa bersama dengan banyak kantor pemerintah. — Maskapai penerbangan membatalkan lebih dari 6.500 penerbangan AS yang dijadwalkan pada hari Selasa, menurut Flightaware. com. — Kota New York bisa mendapatkan salju setebal 20 inci, kata Walikota Bill De Blasio, dengan banjir pesisir dan hembusan angin setinggi 40 hingga 50 mph juga diperkirakan. — Bagian dari Massachusetts bisa melihat 24 inci atau lebih dan angin yang sama kuatnya, kata Gubernur Charlie Baker. — Di Philadelphia, di mana salju diperkirakan mulai turun sekitar pukul 9 malam. m. Senin, akumulasi kemungkinan akan berkisar antara 8 hingga 12 inci, dengan skenario 20 inci kata Samantha Phillips, direktur manajemen darurat kota. "Kami mengimbau masyarakat untuk mempersiapkan diri menghadapi peristiwa yang berdampak besar," katanya. — Di Virginia, Penjaga Pantai menutup Pelabuhan Virginia di pelabuhan di Hampton Roads. Dalam rilis berita Senin malam, badan tersebut mengatakan angin 50 mph yang diprediksi dari badai yang tertunda dapat menciptakan kondisi berbahaya yang akan mempersulit unit Penjaga Pantai untuk menjangkau pelaut yang kesulitan. "Ada bahaya nyata bagi semua kapal di atas air," kata Kapten Rick Wester, kapten pelabuhan. "Saya sangat menyarankan agar para pelaut tetap tinggal di pelabuhan." — "Di mana pun Anda berada saat matahari terbit Selasa pagi, bersiaplah untuk tetap berada di sana selama sisa badai hingga (Selasa) malam," kata Gubernur Connecticut Dannel Malloy. Larangan perjalanan di seluruh negara bagian dijadwalkan mulai berlaku Selasa pukul 5 pagi. m. di seluruh Connecticut. Peringatan dan peringatan badai salju untuk wilayah tersebut, termasuk New Jersey timur laut, New York tenggara dan Connecticut selatan, serta Boston, Philadelphia, Baltimore, dan Washington, dikeluarkan oleh layanan

KODE PROGRAM

Aplikasi pencarian dokumen lintas bahasa (indonesia-inggris) dibuat menggunakan bahasa pemrograman Python 3.9. Ada 6 file kode program yaitu:

1. main.py
2. vector_space_model.py
3. query_expansion.py
4. term_weighting.py
5. docs.py
6. preprocessing.py

Kode sumber dapat diunduh di: <https://github.com/ErwinSputra/CLIR-QE>

main.py

```
import os
```

```
from query_expansion import QE
from preprocessing import Prapengolahan
from vector_space_model import VSM
from docs import Dokumen
from term_weighting import PembobotanKata
import sys
import re
from PyQt5.uic import loadUi
from PyQt5 import QtWidgets, QtGui
from PyQt5.QtWidgets import QMainWindow, QApplication, QTableWidgetItem,
QTableWidgetItem
from PyQt5.QtGui import QPixmap
from PyQt5.QtCore import pyqtSlot, Qt
from deep_translator import GoogleTranslator
import subprocess
```

```
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        loadUi("CLIRwQE.ui", self)
        pixmap = QPixmap('images/AIR&D.jpg')
        self.label.setPixmap(pixmap)
        pixmap = QPixmap('images/logo-unsri.png')
        self.label_2.setPixmap(pixmap)
        self.tabWidget.removeTab(1)
        self.tableWidget.setColumnWidth(1, 610)
        self.parm = 0
        self.tableWidget.move(0, 0)
```

```

        self.tableWidget.setEditTriggers(QTableWidget.NoEditTriggers)
        self.tabWidget.tabCloseRequested.connect(lambda index:
self.tabWidget.removeTab(index))
        self.tableWidget.doubleClicked.connect(self.read_more)
        self.searchBtn.clicked.connect(self.src)
        self.qeBtn.clicked.connect(self.search)
        self.resetBtn.clicked.connect(self.reset)
        self.manualBtn.clicked.connect(self.manual)
        self.exitBtn.clicked.connect(self.exit)

def reset(self):
    print("Reset..")
    if len(self.tabWidget) > 1:
        for tab in range(len(self.tabWidget)-1):
            self.tabWidget.removeTab(1)
    self.qeSystem.setChecked(True)
    self.allres = []
    self.qe0 = []
    self.qe1 = []
    self.qe2 = []
    self.qe3 = []
    self.bool = False
    self.inside_qe = False
    self.allQuery.setChecked(True)
    self.inputQuery.setText("")
    self.sysResult.setPlainText(f"Kueri awal: -\n\nHasil kueri ekspansi
teratas;\n1. -\n2. -\n3. -")
    self.label_1.setText("Hasil Pencarian dengan Kueri - : x artikel")
    self.tableWidget.setRowCount(0)

def manual(self):
    print("Membuka pdf manual..")
    path = 'manual.pdf'
    subprocess.Popen([path], shell=True)

def exit(self):
    print("Exiting..")
    sys.exit()

def src(self):
    self.allres = []
    self.qe0 = []
    self.qe1 = []
    self.qe2 = []
    self.qe3 = []
    self.bool = False
    self.inside_qe = False
    if self.parm == 0:
        self.inv_idx = Dokumen().inverted_index()
        self.tf_idf_doc = PembobotanKata().create_tf_idf()

```

```
self.parm += 1
self.allQuery.setChecked(True)
self.search()
```

```
def search(self):
    query = ""
    if self.allQuery.isChecked():
        query = self.inputQuery.text()
        self.bool = True
    else:
        print("Memilih Kueri Ekspansi")
        self.inside_qe = True
        qe = self.sysResult.toPlainText()
        qe = qe.split("\n")
        new_qe = []
        if self.ogQuery.isChecked():
            self.bool = True
            que = qe[0]
            query = que[12:]

        for qe in qe[3:]:
            new_qe.append(qe[3:])

        if self.qeRes1.isChecked():
            query = new_qe[0]
            if len(self.qe1) != 0:
                self.bool = True
            else:
                self.bool = False
                if new_qe[0] == "-":
                    query = "-"

        elif self.qeRes2.isChecked():
            query = new_qe[1]
            if len(self.qe2) != 0:
                self.bool = True
            else:
                self.bool = False
                if new_qe[1] == "-":
                    query = "-"

        elif self.qeRes3.isChecked():
            query = new_qe[2]
            if len(self.qe3) != 0:
                self.bool = True
            else:
                self.bool = False
                if new_qe[2] == "-":
                    query = "-"
```

```

if query == "":
    msg = QtWidgets.QMessageBox()
    msg.setText("Kueri belum dimasukkan!")
    msg.setIcon(QtWidgets.QMessageBox.Critical)
    msg.exec_()
elif query == "-":
    msg = QtWidgets.QMessageBox()
    msg.setText("Kueri ekspansi tidak ada! Cek kueri pilihanmu")
    msg.setIcon(QtWidgets.QMessageBox.Critical)
    msg.exec_()
else:
    translated_query = GoogleTranslator(source='id',
target='en').translate(query)
    print(translated_query)
    temp = False
    check_res = []
    if self.qeSystem.isChecked() & self.allQuery.isChecked() and
len(self.allres) == 0:
        temp = True
        exp_query, rule = QE().expanding_query(translated_query)
        # exp_query;
        # [winter, wintertime, storm, violent_storm]
        # [pollution, befoulment, defilement, contamination]
        allterm = []
        for term in exp_query:
            if "_" in term:
                output = re.sub(r'_', ' ', term)
                term = output
                allterm.append(term)

        allquery = []
        arr1 = []
        arr2 = []
        arr3 = []
        k = 0
        if rule == 1:
            altres = []
            for item in allterm:
                allquery.append(item)
                q_terms = Prapengolahan().tokenize_and_extract(item)
                print()
                print(q_terms)
                vec_space_model = VSM(self.inv_idx, self.tf_idf_doc)
                res = vec_space_model.cos_sim(q_terms)

                if res != -999:

                    for item in res:
                        if k == 0:
                            self.qe0.append(item)

```

```

        elif k == 1:
            self.qe1.append(item)
        elif k == 2:
            self.qe2.append(item)
        elif k == 3:
            self.qe3.append(item)

        if item[0] not in altres:
            altres.append(item[0])
            self.allres.append(item)

    k += 1
    check_res = self.allres
elif rule == 0:
    arr1 = [allterm[0]]
    arr2 = [allterm[1]]
elif rule == 2:
    arr1 = allterm[0:2]
    arr2 = allterm[2:4]
elif rule == 3:
    arr1 = [allterm[0]]
    arr2 = allterm[1:3]
elif rule == 4:
    arr1 = allterm[0:2]
    arr2 = [allterm[2]]
elif rule == 5:
    arr1 = allterm[0:2]
    arr2 = allterm[2:4]
    arr3 = allterm[4:6]
elif rule == 6:
    arr1 = allterm[0:2]
    arr2 = [allterm[2]]
    arr3 = [allterm[3]]
elif rule == 7:
    arr1 = [allterm[0]]
    arr2 = allterm[1:3]
    arr3 = [allterm[3]]
elif rule == 8:
    arr1 = [allterm[0]]
    arr2 = [allterm[1]]
    arr3 = allterm[2:4]
elif rule == 9:
    arr1 = [allterm[0]]
    arr2 = allterm[1:3]
    arr3 = allterm[3:5]
elif rule == 10:
    arr1 = allterm[0:2]
    arr2 = [allterm[2]]
    arr3 = allterm[3:5]
elif rule == 11:
    arr1 = allterm[0:2]
```

```

        arr2 = allterm[2:4]
        arr3 = [allterm[4]]
    elif rule == 12:
        arr1 = [allterm[0]]
        arr2 = [allterm[1]]
        arr3 = [allterm[2]]
    if rule == 2 or rule == 3 or rule == 4:
        altres = []
        for item1 in arr1:
            for item2 in arr2:
                # "winter storm", "winter violent storm", "wintertime storm",
                "wintertime violent storm"
                que = item1 + " " + item2
                allquery.append(que)
                q_terms = Prapengolahan().tokenize_and_extract(que)
                print()
                print(q_terms)
                # [winter storm], [winter violent storm], [wintertime storm],
                [wintertime violent storm]
                vec_space_model = VSM(self.inv_idx, self.tf_idf_doc)
                res = vec_space_model.cos_sim(q_terms)
                # [23: 0.82, 45: 0.67] atau -999
                if res != -999:
                    for item in res:
                        if k == 0:
                            self.qe0.append(item)
                        elif k == 1:
                            self.qe1.append(item)
                        elif k == 2:
                            self.qe2.append(item)
                        elif k == 3:
                            self.qe3.append(item)

                        if item[0] not in altres:
                            altres.append(item[0])
                            self.allres.append(item)

                    k += 1
                check_res = self.allres
    elif rule >= 5:
        j = 0
        altres = []
        for item1 in arr1:
            for item2 in arr2:
                for item3 in arr3:
                    if j < 4:
                        que = item1 + " " + item2 + " " + item3
                        allquery.append(que)
                        q_terms = Prapengolahan().tokenize_and_extract(que)
                        print()
                        print(q_terms)

```

```

        vec_space_model = VSM(self.inv_idx, self.tf_idf_doc)
        res = vec_space_model.cos_sim(q_terms)
        if res != -999:
            for item in res:
                if k == 0:
                    self.qe0.append(item)
                elif k == 1:
                    self.qe1.append(item)
                elif k == 2:
                    self.qe2.append(item)
                elif k == 3:
                    self.qe3.append(item)

            if item[0] not in altres:
                altres.append(item[0])
                self.allres.append(item)

        j += 1
        k += 1
        check_res = self.allres
        if len(allquery) == 4:
            self.sysResult.setPlainText(
                f"Kueri awal: {allquery[0]}\n\nHasil kueri ekspansi teratas;\n1.
{allquery[1]}\n2. {allquery[2]}\n3. {allquery[3]}")
        elif len(allquery) == 3:
            self.sysResult.setPlainText(
                f"Kueri awal: {allquery[0]}\n\nHasil kueri ekspansi teratas;\n1.
{allquery[1]}\n2. {allquery[2]}\n3. -")
        elif len(allquery) == 2:
            self.sysResult.setPlainText(
                f"Kueri awal: {allquery[0]}\n\nHasil kueri ekspansi teratas;\n1.
{allquery[1]}\n2. -\n3. -")
        else:
            self.sysResult.setPlainText(
                f"Kueri awal: {allquery[0]}\n\nHasil kueri ekspansi teratas;\n1. -
\n2. -\n3. -")

        elif not self.qeSystem.isChecked() and len(self.allres) == 0: # elif
self.radioButton_2.isChecked():
            query_exp = Prapengolahan().tokenize_and_extract(translated_query)
            print()
            print(query_exp)
            vec_space_model = VSM(self.inv_idx, self.tf_idf_doc)
            res = vec_space_model.cos_sim(query_exp)
            # [(('4304', 1.0000000000000002), ('378', 1.0000000000000002),
('4935', 0.6283072553180628)]
            self.allres = res
            check_res = res
            self.qe0 = self.allres

        if res == -999:

```

```

        check_res = []

        if not self.qeSystem.isChecked():
            self.sysResult.setPlainText(
                f"Kueri awal: {query}\n\nHasil kueri ekspansi teratas;\n1. -\n2. -
\n3. -")
        elif self.bool:
            check_res = self.allres
        else:
            check_res = []

        if len(check_res) == 0:
            if self.allQuery.isChecked():
                self.label_1.setText(
                    f"Hasil Pencarian dengan gabungan semua kueri: 0 artikel") #
docs_len / len(res)
            else:
                self.label_1.setText(
                    f"Hasil Pencarian dengan Kueri '{query}': 0 artikel") # docs_len
/ len(res)
            self.tableWidget.setRowCount(0)
            msg = QtWidgets.QMessageBox()
            msg.setText("Artikel Tidak Ditemukan!")
            msg.setIcon(QtWidgets.QMessageBox.Warning)
            msg.exec_()
        else:
            docs_len = 0
            results = []
            qeres = self.allres
            if self.ogQuery.isChecked():
                qeres = self.qe0
            elif self.qeRes1.isChecked():
                qeres = self.qe1
            elif self.qeRes2.isChecked():
                qeres = self.qe2
            elif self.qeRes3.isChecked():
                qeres = self.qe3

            for item in qeres:
                if item[1] >= 0.851:
                    results.append(item)
                    docs_len += 1
            if self.allQuery.isChecked():
                self.label_1.setText(
                    f"Hasil Pencarian dengan gabungan semua kueri: {docs_len}
artikel") # docs_len / len(res)
            else:
                self.label_1.setText(f"Hasil Pencarian dengan Kueri '{query}':
{docs_len} artikel") # docs_len / len(res)

```

```

self.tableWidget.setRowCount(docs_len)

corpus = Dokumenten().document_retr()
allcossim = []
idx = 0
if temp:
    results = sorted(results, key=lambda x: x[1], reverse=True)

self.id_doc = []
self.allnews = []
self.alltitle = []
doc_list = []

for item in results:
    self.id_doc.append(item[0])
    cossim = "%.2f" % item[1]
    allcossim.append(cossim)
    doc = corpus[str(item[0])]
    j = 0
    k = 0
    n = 4000
    doc_news = doc[1]
    for i in range(0, len(doc_news), n):
        if len(doc_news) > n and i + n < len(doc_news):
            while doc_news[i + n + k] != " " and doc_news[i+n+k] != ".":
and doc_news[i+n+k] != "" and doc_news[i+n+k] != "\": #
                k += 1
            doc_list.append(doc_news[i + j:i + n + k])
            j = k

        news = ""
        for text in doc_list:
            translated = GoogleTranslator(source='en',
target='id').translate(text)
            news += translated + " "

        self.allnews.append(news) # news

        doc_title = GoogleTranslator(source='en',
target='id').translate(doc[0])
        self.alltitle.append(doc_title)

    doc_list = []
    idx += 1

idx = 0
for id in self.id_doc:
    item = QTableWidgetItem(str(id))
    item.setTextAlignment(Qt.AlignCenter)
    self.tableWidget.setItem(idx, 0, item)

```

```

        idx += 1
    idx = 0
    for news in self.allnews:
        self.tableWidget.setItem(idx, 1, QTableWidgetItem(news))
        idx += 1
    idx = 0
    for cossim in allcossim:
        cos = str(cossim)
        item = QTableWidgetItem(cos[:6])
        item.setTextAlignment(Qt.AlignCenter)
        self.tableWidget.setItem(idx, 2, item)
        idx += 1
    print("Pencarian selesai..")

@pyqtSlot()
def read_more(self):
    self.new_tab = QtWidgets.QWidget()
    title = ""
    text = ""
    title_tab = ""
    for currentQTableWidgetItem in self.tableWidget.selectedItems():
        title = self.alltitle[currentQTableWidgetItem.row()]
        text = self.allnews[currentQTableWidgetItem.row()]
        title_tab = self.id_doc[currentQTableWidgetItem.row()]

    self.tabWidget.addTab(self.new_tab, f"Dok {title_tab}")
    self.new_layout = QtWidgets.QVBoxLayout(self.new_tab)
    self.new_label = QtWidgets.QLabel(self.new_tab)
    # title = "Kemungkinan penyebab di balik pemboman St. Petersburg"
    self.new_label.setText(title)
    self.new_label.setAlignment(Qt.AlignCenter)
    font = QtGui.QFont()
    font.setPointSize(12)
    font.underline()
    self.new_label.setFont(font)
    self.new_textBrowser = QtWidgets.QTextBrowser(self.new_tab)
    self.textBrowser.setFrameShape(QtWidgets.QFrame.NoFrame)
    self.textBrowser.setReadOnly(True)
    font2 = QtGui.QFont()
    font2.setPointSize(10)
    self.new_textBrowser.setFont(font2)
    self.new_textBrowser.setAlignment(Qt.AlignBaseline)
    self.new_textBrowser.setText(text)
    self.new_layout.addWidget(self.new_label)
    self.new_layout.addWidget(self.new_textBrowser)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    mainWindow = MainWindow()

```

```
widget = QtWidgets.QStackedWidget()
widget.addWidget(mainWindow)
widget.setFixedWidth(985)
widget.setFixedHeight(770)
widget.show()
```

```
try:
    sys.exit(app.exec_())
except:
    print("Exiting..")
```

vector_space_model.py

```
from query_expansion import QE
from term_weighting import PembobotanKata
from preprocessing import Prapengolahan
import math
```

```
class VSM:
```

```
    def __init__(self, inv_idx, tfidf_doc):
        self.query_exp = QE()
        self.inv_idx = inv_idx
        self.tf_idf_doc = tfidf_doc
        self.pmbKata = PembobotanKata()
        self.pra = Prapengolahan()
```

```
    def cos_sim(self, q_expansion):
        tfidf_query = self.pmbKata.tf_idf_query(q_expansion)
        dot_product = 0
        query_mod = 0
        doc_mod = 0
        print(tfidf_query)
        if len(tfidf_query) == 0:
            return -999
        else:
```

```
            cossim = { }
            doks = []
            for term in q_expansion:
                for document in self.inv_idx[term]:
                    if document not in doks:
                        for word in q_expansion:
                            dot_product += tfidf_query.get(word, 0) *
self.tf_idf_doc[word].get(document, 0)
                            query_mod += tfidf_query.get(word, 0) ** 2
                            doc_mod += self.tf_idf_doc[word].get(document, 0) ** 2
                        query_mod = math.sqrt(query_mod)
                        doc_mod = math.sqrt(doc_mod)
                        denominator = query_mod * doc_mod
                        if denominator != 0:
                            cossim[document] = dot_product / denominator
                        dot_product = 0
```

```
        query_mod = 0
        doc_mod = 0
        doks.append(document)
    return sorted(cossim.items(), key=lambda x: x[1], reverse=True)
```

query_expansion.py

```
from nltk.corpus import wordnet as wn
from preprocessing import Prapengolahan
import numpy as np
```

```
class QE:
```

```
    def __init__(self):
        self.pra = Prapengolahan()
```

```
    def expanding_query(self, query):
        query = self.pra.tokenize_and_extract(query)
        rule = -1
        idx = 1
        if len(query) == 1:
            idx = 3
            rule = 1
        elif len(query) == 2:
            rule = 2
        elif len(query) == 3:
            rule = 5
```

```
        dict = {}
        synonyms = []
        for term in query:
            i = 0
            dict[term] = False
            synonyms.append(term)
            if len(wn.synsets(term)) != 0:
                for syn in wn.synsets(term):
                    for lem in syn.lemmas():
                        if lem.name().lower() not in synonyms and i != idx: #
lem.name().lower()
                            synonyms.append(lem.name())
                            dict[term] = True
                            i += 1
```

```
        theterm = ""
        testterm = []
        i = 0
        if len(dict) > 1:
            for item in dict:
                if not dict[item]:
                    if rule == 2:
                        theterm = item
                    if rule == 5:
```

```

        testterm.append(item)
        i += 1

    if rule == 5:
        if i == 3:
            rule = 12
        elif len(testterm) == 2:
            main_term = np.setdiff1d(query, testterm)
            index = query.index(main_term)
            if index == 0:
                rule = 6
            if index == 1:
                rule = 7
            if index == 2:
                rule = 8
        elif len(testterm) == 1:
            index = query.index(testterm[0])
            if index == 0:
                rule = 9
            if index == 1:
                rule = 10
            if index == 2:
                rule = 11

    if rule == 2:
        if theterm != "":
            index = query.index(theterm)
            if index == 0:
                rule = 3
            if index == 1:
                rule = 4
        elif i == 2:
            rule = 0

    return synonyms, rule

@staticmethod
def list_to_string(s):
    str1 = ""
    i = 1
    for ele in s:
        if i == len(s):
            str1 += ele
        else:
            str1 += ele + " "
        i += 1

    return str1

```

term_weighting.py

from docs import Dokumen

```
import math
from collections import defaultdict
```

```
class PembobotanKata:
```

```
    def __init__(self):
        self.docs = Dokumen().documents_dictionary()
        self.N = len(self.docs)
        self.avg_len = sum([len(doc) for doc in self.docs.values()])/len(self.docs)
        self.inv_idx = Dokumen().inverted_index()
```

```
    def tf_idf_query(self, q_terms):
        fqt = { }
        for term in q_terms:
            fqt[term] = fqt.get(term, 0) + 1
        tf_idf_query = { }
        id = 1
        for term in fqt.keys():
            query_tf = math.log10(fqt[term]) + 1
            print(f'{id}. {term}')
            df = len(self.inv_idx[term])
            print(f'df = {df}')
            if df != 0:
                query_idf = math.log(self.N / df)
                id += 1
                tf_idf_query[term] = query_tf * query_idf
        return tf_idf_query
```

```
    def tf_idf_doc(self, term, docID):
        td = self.docs[docID].count(term)
        df = len(self.inv_idx[term]) # df is the number of documents a term occurs
```

```
in
```

```
    tf = math.log10(td) + 1 # the frequency of the word t in document d
    idf = math.log(self.N/df)
    w = tf * idf
    return w
```

```
    def create_tf_idf(self):
        tf_idf = defaultdict(dict)
        for term in set(self.inv_idx.keys()):
            for docid in self.inv_idx[term]:
                tf_idf[term][docid] = self.tf_idf_doc(term, docid)
        return tf_idf
```

docs.py

```
from preprocessing import Prapengolahan
from collections import defaultdict
```

```
class Dokumen:
```

```
    def __init__(self):
```

```

self.Development_Docs = "dataset/demofile.dataset" # devdocs.dataset
demofile.dataset
self.Output_Docs = "dataset/doc_retr.dataset"
self.pra = Prapengolahan()
self.documents_dict = {}

def documents_dictionary(self):
    file = open(self.Development_Docs, encoding="UTF-8") # ISO-8859-1
    for line in file:
        doc = line.split("\t")
        terms = self.pra.tokenize_and_extract(doc[1])
        self.documents_dict[doc[0]] = terms
    file.close()

    return self.documents_dict

def inverted_index(self):
    self.documents_dictionary()
    inverted_index = defaultdict(set)

    for docid, terms in self.documents_dict.items():
        for term in terms:
            inverted_index[term].add(docid)
    return inverted_index

def document_retr(self):
    documents = {}
    file = open(self.Output_Docs, encoding="UTF-8")

    for line in file:
        doc = line.split("\t")
        text = [doc[1], doc[2]]
        documents[doc[0]] = text
    file.close()

    return documents

```

preprocessing.py

```

from nltk.tokenize import word_tokenize
import re
import nltk

class Prapengolahan:
    def __init__(self):
        self.tokenizer = word_tokenize
        self.stopwords = set(nltk.corpus.stopwords.words("english"))
        self.lemma = nltk.wordnet.WordNetLemmatizer()

    def tokenize_and_extract(self, doc):
        doc = doc.lower()

```

```
tokenize = [token for token in self.tokenizer(doc)]

terms = []
self.stopwords.add('self')
for token in tokenize:
    if token not in self.stopwords:
        # jika token bukan angka dan bukan selain huruf A-Z sama dengan
        # jika token hanya mengandung huruf A-Z, selain itu not accept
        if not re.search(r'\d', token) and not re.search(r'[^\A-Za-z-]', token):
            lem = self.lemma.lemmatize(token)
            terms.append(lem)
return terms
```

BUKU MANUAL

Aplikasi Pencarian Dokumen Lintas Bahasa (Indonesia-Inggris)

Alat Bantu Penelitian Untuk Bidang Cross-Language Information Retrieval (CLIR) Dengan
Menggunakan Query Expansion (QE)

Pengembang



Aplikasi Versi 1.0.0 © AIRD-UNSRI-2023

Buku Manual – Versi 1.0 (Januari 2023)

DAFTAR ISI

DAFTAR ISI

A. APLIKASI PENCARIAN DOKUMEN LINTAS BAHASA (INDONESIA-INGGRIS) ...	1
1.1 Tentang Aplikasi	1
1.2 Pengembang	1
B. INSTALASI APLIKASI	2
C. TAMPILAN APLIKASI	3
3.1 Antarmuka Utama	3
3.2 Antarmuka untuk Membaca	5
D. TUTORIAL APLIKASI	6
E. KONTAK DAN SARAN	9
5.1 Kontak	9
5.2 Saran	9

A. APLIKASI PENCARIAN DOKUMEN LINTAS BAHASA (INDONESIA-INGGRIS)

1.1 Tentang Aplikasi

Cross-Language Information Retrieval (CLIR) atau Pencarian Dokumen Lintas Bahasa adalah sebuah sistem yang berfungsi untuk mengambil sebuah informasi yang ditulis dalam bahasa yang berbeda dari bahasa permintaan pengguna. Misalnya pada aplikasi penelitian ini, pengguna menggunakan kueri dalam bahasa Indonesia, tetapi informasi relevan yang diambil adalah dalam bahasa Inggris.

Query Expansion (QE) adalah sebuah teknik yang digunakan dalam sistem *Information Retrieval* (IR) untuk meningkatkan kinerja pengambilan terutama pada sisi *recall*, dengan memformulasi ulang kueri dan melakukan penelusuran kedua untuk mendapatkan dokumen relevan lainnya yang tidak dapat diambil oleh kueri awal pengguna.

Aplikasi Pencarian Dokumen Lintas Bahasa menggunakan QE merupakan aplikasi yang dibangun guna membantu penelitian untuk bidang *Cross-Language Information Retrieval* (CLIR) dengan menggunakan *Query Expansion* (QE). Adapun 1 komponen penting lainnya pada aplikasi ini yaitu *Cosine Similarity*. *Cosine Similarity* merupakan salah satu pengukur kemiripan teks yang sering digunakan pada metode *Vektor Space Model* dalam sistem pencarian dokumen. *Cosine Similarity* digunakan untuk menghitung nilai cosinus sudut antara dua vektor yaitu vektor dokumen dan vektor kueri. Tiap vektor tersebut mempresentasikan setiap kata dalam setiap dokumen (teks) yang dibandingkan dan membentuk sebuah segitiga. Jika dua vektor yang dibandingkan identik, maka sudutnya adalah 0° dan nilai kesamaannya adalah 1; dan jika dua vektor yang dibandingkan tidak identik sama sekali, maka sudutnya adalah 90° dan nilai kesamaannya adalah 0.

1.2 Pengembang

Erwin Saputra

Dr. Abdiansah, S.Kom., M.CS.

Novi Yusliani, S.Kom., M.T.

Fakultas Ilmu Komputer - Universitas Sriwijaya

2023

B. INSTALASI APLIKASI

Instalasi aplikasi saat ini hanya bisa dilakukan secara manual karena belum tersedia software instalasi. Untuk meminimalisir terjadinya *error* pada proses instalasi dan eksekusi program, sebaiknya membangun *environment* dengan versi yang sama. Ada beberapa *software* yang perlu diinstall sebelum menginstall aplikasi, yaitu:

- Python 3.9
- Pycharm Community Edition 2020 3.2

Setelah melakukan proses instalasi *environment* tersebut. Langkah pertama yang harus dilakukan ialah mengunduh kode sumber di Github: <https://github.com/ErwinSputra/CLIR-QE>

Langkah selanjutnya ialah melakukan instalasi kode sumber. Berikut langkah-langkahnya:

1. Membuka folder kode sumber di aplikasi Pycharm dengan cara klik tombol "File" di kiri atas menu utama Pycharm, yang kemudian dilanjutkan dengan menekan tombol Open. Lalu pilihlah folder kode sumber yang telah diunduh sebelumnya. Dan klik OK

2. Install pustaka-pustaka pendukung yang terdapat di file "requirement.txt". File ini dapat ditemukan di dalam folder CLIR-QE yang telah diunduh. Hal ini dapat dilakukan dengan cara

- "Open File > Settings > Project".
- Kemudian klik tab "Python Interpreter" di dalam tab.
- Lalu klik simbol + kecil untuk menambahkan pustaka baru ke proyek.
- Sekarang ketik pustaka yang akan diinstall, misalnya PyQt5, dan klik "Install Package".
- Tunggu instalasi untuk menghentikan dan menutup semua jendela popup. Setelah selesai, lanjutkan untuk menginstall pustaka-pustaka lain yang dibutuhkan sesuai dengan instruksi sebelumnya.

3. Jalankan aplikasi dengan klik tombol *play* berwarna hijau ► di tengah kanan atas. Atau bisa juga dengan mengklik tombol "► Run.." yaitu dengan cara mengklik kanan mouse pada *file* main.py.

Jika ada kendala saat proses instalasi, silakan buka *issue* di Github¹ supaya bisa dibantu oleh pembuat atau pengguna lain.

C. TAMPILAN APLIKASI

3.1 Antarmuka Utama

Berikut adalah tampilan antarmuka utama aplikasi Pencarian Dokumen Lintas Bahasa (CLIR) Indonesia-Inggris setelah dijalankan.

python

Main tab x Dok 1730 x

☒ Sistem QE

Cross-Language Information Retrieval

badai salju

Cari

Kueri awal: snowstorm

Hasil kueri ekspansi teratas;

1. blizzard
2. -
3. -

☐ Semua kueri
☐ Kueri awal
☒ Kueri ekspansi 1
☐ Kueri ekspansi 2
☐ Kueri ekspansi 3

Hasil kueri

Hasil Pencarian dengan Kueri 'blizzard': 4 artikel

No Dokumen	Artikel	Cos Similarity
1 1747	(CNN) Sebuah nor'easter sebagian besar terhindar dari New York City dan ...	1.00
2 1730	(CNN) Dengan badai salju yang meluncur ke arah Timur Laut, wilayah ...	1.00
3 8565	(CNN) Sersan. Jennifer Bruno tahu badai salju yang melanda New England ...	1.00
4 8549	(CNN) Badai salju besar pertama tahun 2015 telah melanda Amerika Serika...	1.00

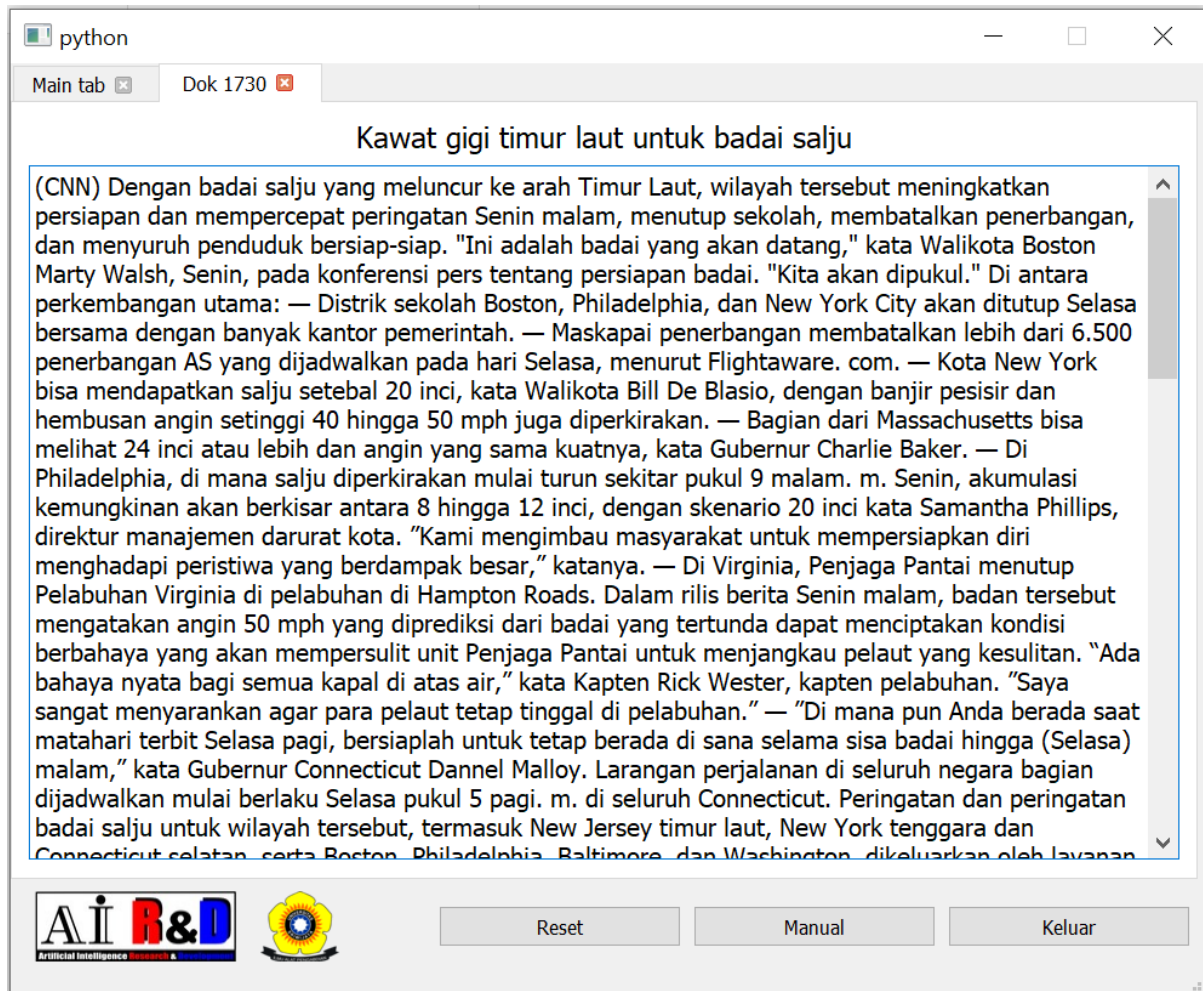
Reset Manual Keluar

Tabel 1. Deskripsi komponen-komponen aplikasi

No	Nama Komponen	Deskripsi
1.	Checkbox [Sistem QE]	Komponen ini berfungsi untuk menentukan sistem yang akan digunakan. Secara <i>default</i> , checkbox akan tercentang yang menandakan bahwa sistem akan menggunakan QE, sebaliknya jika pengguna mengosongkan centang di checkbox ini maka sistem tidak akan menggunakan QE.
2.	Form isian kueri	Komponen ini menerima masukan kueri berupa teks yang maksimal berisi 3 kata saja dalam bahasa Indonesia dari pengguna.
3.	Tombol [Cari]	Komponen ini harus diklik ketika pengguna sudah mengisi Form isian kueri yang berfungsi untuk memproses kueri yang dimasukkan tersebut.
4.	Hasil kueri ekspansi	Komponen ini adalah sebuah tampilan hasil kueri ekspansi yang telah didapatkan dari sistem.
5.	Radio Button [Hasil kueri ekspansi]	Komponen ini berfungsi untuk memilih kueri mana yang ingin ditampilkan hasilnya. “Semua kueri” secara <i>default</i> yang akan ditampilkan. “Semua kueri” sendiri akan menampilkan hasil gabungan dari semua kueri.
6.	Tombol [Hasil Kueri]	Komponen ini harus diklik ketika pengguna sudah menentukan kueri mana yang ingin ditampilkan.
7.	Label Informasi	Sebuah teks informasi mengenai kueri yang digunakan beserta jumlah artikel yang ditemukan.
8.	Tabel hasil pencarian	Sebuah tabel hasil pencarian sistem yang terdiri dari 3 kolom yaitu “No Dokumen”, “Artikel” dan “Cos Similarity”.
9.	Tombol operasi standar	Ada tiga tombol operasi standar, yaitu: tombol [Reset], [Manual], [Keluar]
10.	Tab baru	Sebuah tab baru yang muncul setelah pengguna mengklik salah satu sel yang ada di Tabel hasil pencarian. Seperti pada gambar, pengguna mengklik sel di kolom “Artikel” pada baris kedua sehingga muncullah tab baru ini yang berfungsi untuk membaca artikel.

3.2 Antarmuka untuk Membaca

Selanjutnya ialah antarmuka untuk membaca pada aplikasi Pencarian Dokumen Lintas Bahasa (CLIR) Indonesia-Inggris. Seperti yang terlihat, pada antarmuka ini terdapat judul dan isi dari artikel yang ingin dibaca.



D. TUTORIAL APLIKASI

Langkah 1:

- Buka aplikasi Pencarian Dokumen Lintas Bahasa (CLIR)
- Pilih antara sistem QE atau Non QE. Jika ingin sistem Non QE, kosongkan checkbox [Sistem QE]. Sebaliknya jika ingin menggunakan sistem QE, checkbox diabaikan saja karena secara *default* sistem QE yang digunakan.
- Masukkan kueri seperti, “Badai salju”. Maksimal 3 kata.
- Tekan tombol [Cari]. Tunggu beberapa menit..

python

Main tab

☒ Sistem QE

Cross-Language Information Retrieval

badai salju Cari

Kueri awal: snowstorm

Hasil kueri ekspansi teratas;

1. blizzard
2. -
3. -

☒ Semua kueri
☐ Kueri awal
☐ Kueri ekspansi 1
☐ Kueri ekspansi 2
☐ Kueri ekspansi 3

Hasil kueri

Hasil Pencarian dengan gabungan semua kueri: 7 artikel

No Dokumen	Artikel	Cos Similarity
1 937	(CNN) New York dan Boston akan menutup sekolah umum mereka pada ...	1.00
2 966	(CNN) Timur Laut dihantam Kamis dengan apa yang tampaknya menjadi ...	1.00
3 8528	(CNN) Ini bukan badai salju biasa, kata pakar cuaca. Layanan Cuaca Nasion...	1.00
4 8549	(CNN) Badai salju besar pertama tahun 2015 telah melanda Amerika Serika...	1.00
5 1730	(CNN) Dengan badai salju yang meluncur ke arah Timur Laut, wilayah ...	1.00
6 1747	(CNN) Sebuah nor'easter sebagian besar terhindar dari New York City dan ...	1.00

Reset Manual Keluar

Langkah 2:

- Hasil yang ditampilkan adalah hasil dari gabungan semua kueri. Sehingga jika ingin menampilkan salah satu kueri saja, dapat dilakukan dengan memilih terlebih dahulu kueri mana yang ingin ditampilkan pada Radio Button [Hasil kueri ekspansi]. Contohnya dengan memilih “Kueri ekspansi 1”.
- Kemudian tekan tombol [Hasil kueri]. Tunggu beberapa saat..

python

Main tab

☒ Sistem QE

Cross-Language Information Retrieval

badai salju Cari

Kueri awal: snowstorm

Hasil kueri ekspansi teratas;

1. blizzard

2. -

3. -

☐ Semua kueri
☐ Kueri awal
☒ Kueri ekspansi 1 Hasil kueri
☐ Kueri ekspansi 2
☐ Kueri ekspansi 3

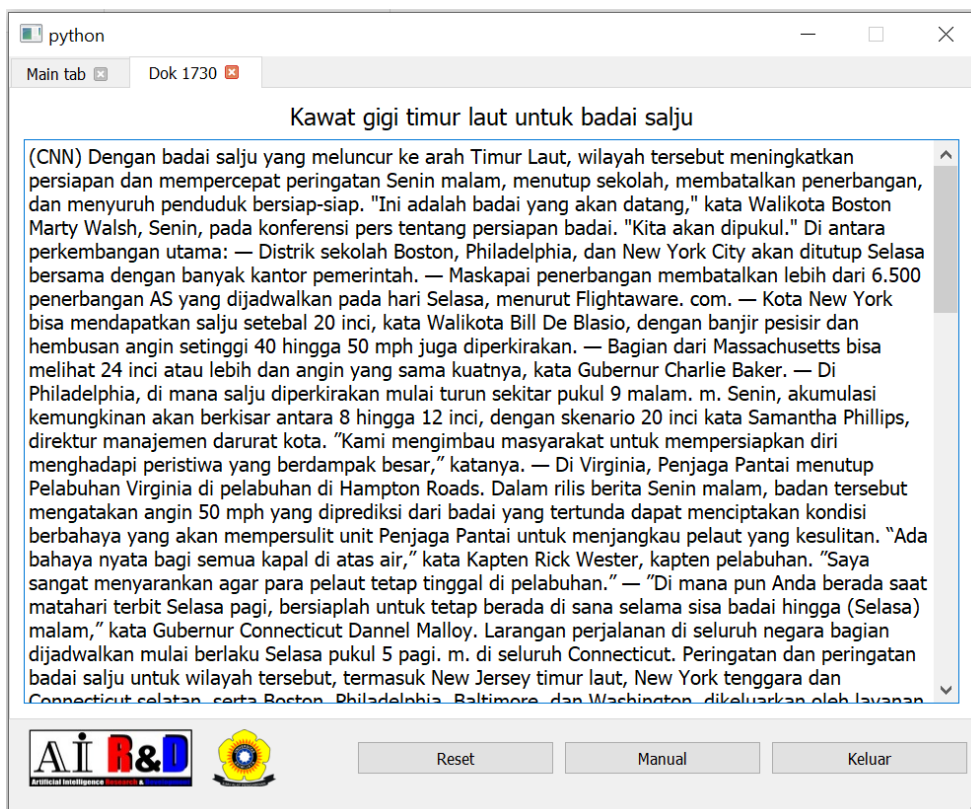
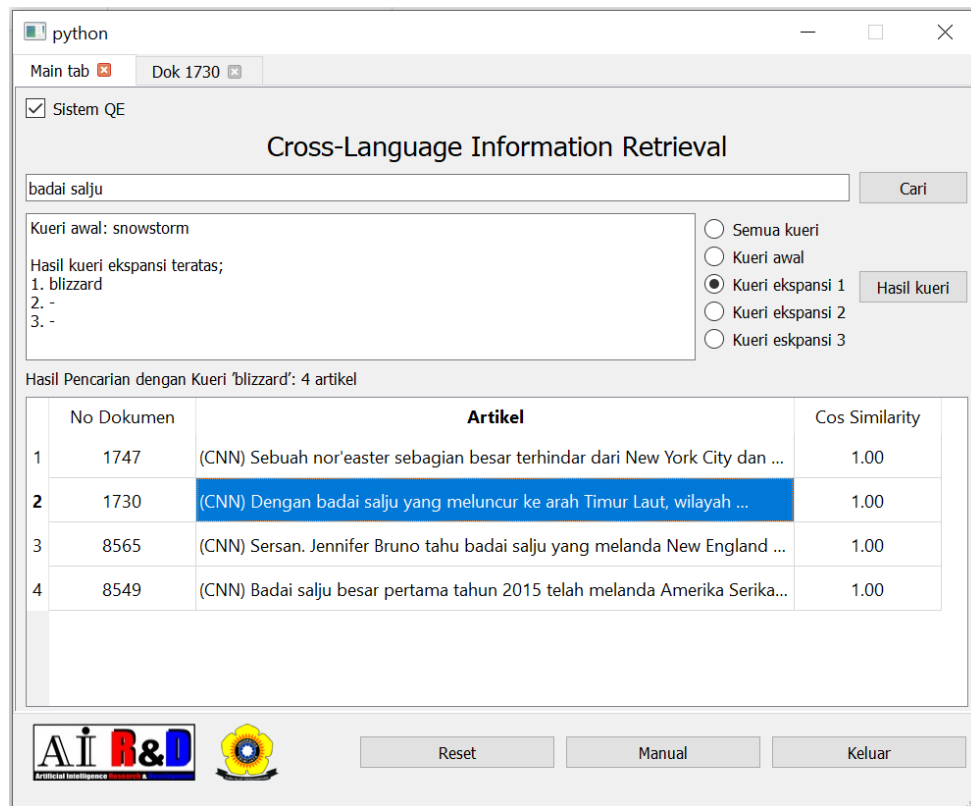
Hasil Pencarian dengan Kueri 'blizzard': 4 artikel

	No Dokumen	Artikel	Cos Similarity
1	8549	(CNN) Badai salju besar pertama tahun 2015 telah melanda Amerika Serika...	1.00
2	1730	(CNN) Dengan badai salju yang meluncur ke arah Timur Laut, wilayah ...	1.00
3	1747	(CNN) Sebuah nor'easter sebagian besar terhindar dari New York City dan ...	1.00
4	8565	(CNN) Sersan. Jennifer Bruno tahu badai salju yang melanda New England ...	1.00

Reset Manual Keluar

Langkah 3:

- Untuk membaca sebuah artikel, tinggal klik pada salah satu bagian tabel (sel) pada artikel yang ingin dibaca. Seperti contoh pada bagian kolom “Artikel” baris kedua.



E. KONTAK DAN SARAN

5.1 Kontak

Kontak pembuat aplikasi dapat dihubungi di alamat berikut ini:

AIRD (Artificial Intelligence **R**esearch & **D**evelopment)
Laboratorium Kecerdasan Buatan & Grafika Komputer
Fakultas Ilmu Komputer, Universitas Sriwijaya
CP: erwin saputra / 09021281823170@student.unsri.ac.id

Video demo program dapat diakses di

<https://youtu.be/hIDLtV40bhg>

5.2 Saran

Saran dan masukan anda dapat dikirimkan via email:

09021281823170@student.unsri.ac.id

Atau anda bisa langsung berpartisipasi melalui Github:

<https://github.com/ErwinSputra/CLIR-QE>