Solusi Pendekatan SAT Problem dengan Jaringan Syarf Tiruan Model Learning Vector Quantization

Luna Bunga Karolina
Program Studi Informatika – Program Sarjana
Universitas Islam Indonesia
Yogyakarta, Indonesia
17523116@students.uii.ad.id

Taufiq Hidayat
Program Studi Sarjana Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
taufiq.hidayat@uii.ac.id

***Abstract*—Boolean Satisfiability Probelm atau yang sering disingkat dengan SAT Problem merupakan suatu masalah yang menentukan sebuah formula matematika bernilai satisfiable atau unsatisfiable. Sebuah perangkat lunak bagi solusi untuk permasalah SAT disebut dengan SAT Solver. Salah satu metode yang dapat digunakan untuk membuat SAT Solver adalah Jaringan Syaraf Tiruan (JST) atau dalam bahasa inggrisnya disebut Artificial Neural Network (ANN). JST ini memiliki banyak model, namum pada penelitian ini menggunakan JST dengan model Learning Vector Quantization (LVQ). SAT Solver ini akan dibangun dengan menggunakan bahasa pemograman Python yang merupakan bahasa pemograman tingkat tinggi. Data yang digunakan dalam penelitian ini menggunakan format Conjuctive Normal Form (CNF) dan data tersebut kemudian melalui proses pelatihan agar mendapatkan luaran berupa penentuan satisfiable atau unsatisfiable. Hasilnya adalah sistem mampu membaca input file CNF yang masuk kemudian menguraikannya menjadi pasangan input dan output. Selanjutkan data tersebut melalui proses pelatihan sehingga pada akhirnya mampu mengeluarkan output yang berisi *string* satisfiable atau unsatisfiable.**

Keywords—SAT Problem, SAT Solver, Jaringan Syaraf Tiruan, Learning Vector Quantization

I. PENDAHULUAN

Manuisa diciptakan dengan bentuk yang paling sempurna oleh Tuhan Yang Maha Esa. Bentuk yang sempurna ini mendorong para ilmuan untuk mempelajari struktur dan cara kerja manusia kemudian diterapkan ke dalam teknologi. Salah satu hasil implementasi dari fungsi manusia adalah Jaringan Syaraf Tiruan (JST) atau dalam bahasa inggrisnya disebut dengan Artificial Neural Network (ANN).

Jaringan Syaraf Tiruan pertama kali diperkenalkan pada tahun 1943 oleh McCulloch dan Pitts. Sejak saat itu JST terus mengalami perkembangan sampai saat ini. JST ini memiliki berbagai macam model yang bisa disesuaikan dengan kebutuhan dan tujuan yang ingin dicapai oleh *developer*. Macam-macam model JST diantaranya adalah backpropagation, Graph Neural Network, Convolution Neural Network, K-SOM, dan Learning Vector Quantization. Implementasi dari berbagai model JST ini bisa dilihat pada memprediksi tingkat polusi udara, meramalkan beban listrik jangka pendek pada perencanaan distribusi listrik, membuat model prediksi kadar salinitas air sungai menggunakan metode jaringan syaraf tiruan, dan menggunakan jaringan syaraf tiruan untuk memprediksi curah hujan dan klimatologi [6].

Di sisi lain, dalam bidang matematika dan informatika kita mengenal istilah Boolean Satisfiability Problem atau yang disingkat dengan SAT Problem, yang merupakan salah satu bidang dalam logika matematika. Tujuan akhir dari SAT Problem ini adalah menentukan sebuah formula matematika yang bernilai satisfiable atau unsatisfiable. Masalah yang bisa diselesaikan menggunakan penyelesaian SAT ini adalah untuk menyelesaikan masalah-masalah kecerdasan. Sebuah perangkat lunak untuk menyelesaikan SAT Problem disebut dengan SAT Solver. Contoh aplikasi SAT Problem dalam kehidupan sehari-hari adalah pada Sudoku[12].

Metode yang bisa digunakan untuk membuat SAT Solver adalah JST. Penelitian sebelumnya yang menggunakan JST dalam menyelesaikan SAT Solver adalah penelitian yang dilakukan Alexander yang merancang SAT Solver menggunakan model K-SOM [14]. Namun penelitian ini masih memiliki kekurangan, yaitu SAT Solver yang tawarkan belum memberikan solusi yang sepenuhnya memuaskan. Akurasinya lebih tinggi ketika hanya berlaku untuk masalah-masalah besar tetapi tidak berlaku untuk masalah yang kecil. Meski memiliki kekurangan, dari penelitian-penelitian sebelumnya bisa dilihat bahwa Jaringan Syaraf Tiruan memang memiliki kemampuan untuk menyelesaikan masalah-masalah dengan kompleksitas yang tinggi.

Oleh karena itu dalam penelitian ini peneliti ingin memberikan solusi lain berupa perangkat lunak yang bisa menyelesaikan SAT Problem. SAT Solver yang dikembangkan dalam penelitian ini dirancang menggunakan Jaringan Syaraf Tiruan model Learning Vector Quantization (LVQ). Diharapkan dengan menggunakan model LVQ ini solusi yang diberikan lain bagi permasalah SAT Problem dengan model yang lebih baik dari sebelumnya.

II. LANDASAN TEORI

A. *Tinjauan Pustaka*

Penelitian SAT Solver ini sudah banyak dilakukan sebelumnya, diantaranya sebagai berikut.

SAT Problem merupakan suatu problem yang menentukan apakah sebuah formula akan bernilai true atau false. SAT solver merupakan suatu sistem yang mampu menyelesaikan SAT Problem. Sistem ini bisa dibangun dengan berbagai metode namun pada penelitian ini akan difokuskan menyelesaikan SAT Problem menggunakan Jaringan Syaraf Tiruan dengan model Learning Vector Quantization. Beberapa penelian terkait topic ini diantaranya adalah penelitian oleh Daniel Selsam membahas tentang penyelesain SAT Problem menggunakan jaringan syaraf. Penelitian ini membuktikan kemampuan jaringan syaraf sederhana untuk menyelesaikan masalah SAT Problem yang kompleks menggunakan model Graph Neural Netwok (GNN). Penelitian ini membuktikan bahwa jaringan syaraf yang sederhana pun bisa menyelesaikan masalah SAT yang kompleks.

Penelitian selanjutnya dari Setiawan yang meneliti SAT Solver menggunakan pendekatan algoritma genetika. Penelitian ini membangun SAT Solver dengan metode pendekatan dan mencapai hasil yang cukup memuaskan dengan keberhasilan menyelesaikan SAT Problem dengan hitungan detik [16]. Namun penelitian ini masih memiliki kekurangan karena SAT Problem yang diuji pada penelitian masih terlalu sederhana sehingga diperlukan penelitian lebih lanjut untuk mendapatkan SAL Solver yang lebih baik.

Penelitain lain juga dilakukan oleh Pratama yang membahas mengenai penyelesai SAT Problem menggunakan algoritma DPLL. Hasil yang diperoleh adalah waktu yang SAT Solver ini butuhkan untuk menyelesaikan SAT Problem tidak dipengaruhi oleh klausa. Selain itu sistem yang dibangun dapat menyelesaikan permainan Sudoku dengan konsep yang telah dibangun [15]. Kekurangan dari penelitian ini adalah dari segi User Interface yang belum ada.

B. Dasar Teori

Teori-teori yang digunakan sebagai dasar penelitian ini dirangkum sebagai berikut:

a) SAT Problem dan SAT Solver

Boolean Satifiability Problem atau yang disingkat dengan SAT Problem adalah masalah fundamental dalam ilmu computer yang akan menentukan suatu masalah memenuhi rumus Boolean tertentu [13]. SAT Problem ini akan menanyakan apakah ada assignment yang memuasakan dalam variable rumus preposisi. Jika terdapat nilai yang memuaskan dalam rumus tersebut, maka masalah tersebut akan dikatakan satisfiable atau SAT. Jika tidak, maka masalah tersebut akan dinyatakan unsatisfiable atau UNSAT [3]. Masalah SAT ini akan melibatkan satu set variable Boolean x_1, x_2, ... x_n dengan rumus F berbentuk format Conjunctive Normal Form (CNF). Rumus ini merupakan gabungan dari m dari klausa c_1, c_2, c_3, … c_m. Klausa adalah disjungsi dari satu atau lebih literal, di mana literal adalah variabel xn atau negasinya. Rumus F memuaskan jika ada bisa ditetapkan kebenaran untuk setiap variabelnya memenuhi setiap klausa rumus. Jika tidak, maka rumus tersebut tidak memuaskan. Tujuan dari penelitian ini adalah untuk menentukan assignment untuk setiap variabel x yang memenuhi semua klausa [1]. Ekspresi Boolean yang dibuat menggunakan konstanta true (1) dan false (0), variabel, negasi, konjungsi, dan disjungsi disebut rumus logika proposisional. Model untuk rumus adalah penugasan nilai Boolean ke variabelnya sedemikian rupa sehingga rumus mengevaluasi ke 1 (true). Untuk setiap rumus, terdapat rumus equisatisfiable dalam bentuk normal konjungtif (CNF), yang dinyatakan sebagai konjungsi dari disjungsi variabel (mungkin dinegasikan). Setiap konjungsi rumus di CNF disebut klausa, dan setiap variabel (mungkin dinegasikan) dalam klausa disebut literal [4].

b) Jaringan Syaraf Tiruan

Jaringan neural adalah sekelompok node yang saling berhubungan, mirip dengan jaringan neuron yang luas di otak manusia [6]. Jaringan Syaraf Tiruan sederhana awalnya dikenalkan oleh McCulloch dan Pits (1943). Selanjutnya Rosebalt (1958) mulai memperkenalkan dan mengembangkan model jaringan yang disebut dengan *Perceptron*. Modetode pelatihan pada jaringan ini mengoptimalkan hasil dari iterasinya [2].

Jaringan Syaraf Tiruan merupakan paradigm pemrosesan suatu informasi yang terinspirasi oleh sistem sel saraf biologi. Elemen dasar dari paradigma tersebut adalah struktur yang baru dari proses pemrosesan informasi tersebut. Sama seperti seperti manusia, Jaringan Syaraf Tiruan belajar dari contoh. Jaringan Syarf Tiruan dibentuk untuk menyelesaikan suatu masalah tertertetu seperti pengenalan pola atau klasifikasi karena proses pembelajaran [7].

Jaringan syaraf juga terdiri dari beberapa neuron seperti halnya otak manusia. Neuron-neuron tersebut akan mentransformasikan informasi yang diterima melalui sambungan keluarnya menuju ke *neuron-neuron* yang lain. Pada JST ini dikenal dengan nama bobot. Dalam bobot itu lah jaringan syaraf akan menyimpan informasi ilmu pengetahuannya [6].

Elemen pemrosesan yang bernama neuron tersebut, disebut juga degan cell atau node. Setiap neuron akan mengaktifkan fungsi aktivasi terhadap input jaringan. Proses pengolahan informasi pada JST terjadi pada *neuron* dan disalurkan melalui *link* yang saling berhubungan dan memiliki bobot [12].

c) Conjunctive Normal Form

Bentuk Normal (Normal Form) adalah ekspresi logika yang disusun sedemikian rupa sehingga hanya berisikan variable-variabel proposisional maupun negasinya. Perangkai dasar atau penguhubung yang digunakan adalah AND dan OR. Tetapi jika disebut dengan Bentuk Nornal Kunjungtif atau yang bahasa inggrisnya disebut Conjunctive Normal Form, maka proposisi majemuk di dalam satu kurung dengan kurng lainnya dirangkai dengan dengan AND dan proposisi manjemuk di dalam kurung dirangkai dengan OR [8].

1. Hanya memuat operator alami, yaitu ∧, ∨, dan ¬.
2. Operator ¬ hanya boleh diterapkan terhadap proposisi (sub-formula yang berupa formula atomik)

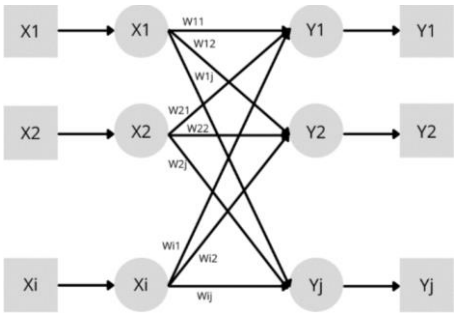
CNF adalah salah satu bentuk formula yang termasuk formula dalam normal form karena memenuhi kedua syarat tersebut. Formula-formula berikut merupakan formula dalam bentuk CNF. Untuk memudahkan pembacaan, setiap klausa yang terdiri dari lebih 1 literal ditulis dalam tanda kurung. Contoh CNF adalah [11]:

- a ∧ b
- a ∧ (b ∨ ¬c)
- a ∧ b ∧ (¬a ∨ ¬b)
- (a ∨ ¬b) ∨ (c ∨ ¬d)
- (a ∨ ¬b) ∧ b ∧ (¬a ∨ ¬c)
- (a ∨ ¬b) ∨ (c ∨ ¬d) ∧ (b ∨ d)
- (a ∨ ¬b) ∧ (b ∨ ¬c) ∧ (c ∨ ¬a)

d) Learning Vector Quantization

Menurut Kusumadewi (2013), *Learning Vector Quantization* (LVQ) adalah suatu metode untuk melakukan pembelajaran pada lapisan kompetitif yang terawasi. Lapisan kompetitif pada LVQ akan secara otomatis belajar untuk mengklasifikasinya vektor *input*. Kelas yang sudah didapat sebagai hasil dari lapisan kompetitif ini bergantung pada jarang antara vektor-vektor *input*. Jika vektor ada sekumpulan vektor yang mendekati sama, maka lapisan kompetitif akan meletakkan kedua vektor tersebut ke dalam satu kelas yang sama [10].

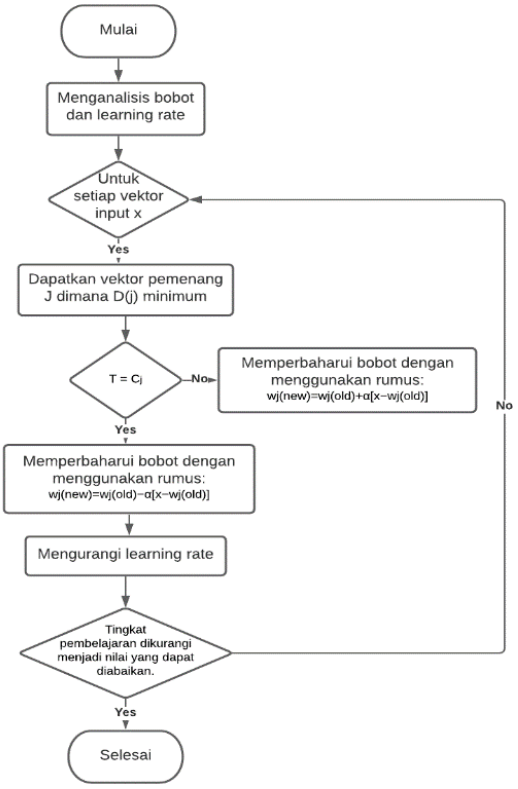
Pada tahun 1986, Kohonen mengusulkan model pembelajaran LVQ ini sebagai perbaikan dari Vector Quantization. Tujuan dari algoritma ini adalah untuk meminimalisasi kesalahan dalam pengklasifikasian. LVQ merupakan metode klasifikasi yang setiap setiap unit outputnta merepresentasikan sebuah kelas. Karena termasuk ke dalam kelompok yang terawasi, LVQ digunakan untuk pengelompokkan di mana jumlah kelompok sudah ditentukan arsitekturnya. *Kohonen-Self Organizing Map* (K-SOM) dan LVQ sebenarnya mempunyai arsitektur yang mirip karena LVQ merupakn versi algoritma jaringan syaraf terawasi dari K-SOM. Model pembelajaran ini dirancang untuk melakukan proses pelatihan lebih cepat dibandingkan *Backpropagation*. Hal ini dapat meringkas atau mengurangi *dataset* besar untuk sejumlah kecil vektor [9].



Gambar 1. Arsitektur jaringan LVQ

Gambar di atas menjelaskan arsitektur dari JST dengan model LVQ. Jika dilihat dari modelnya, dapat dilihat bahwa ada *input* yang diberi nama X dan *output* Y. masing-masing dari *input* ini memiliki bobot (*weight*) yang terkoneksi ke semua *output* yang ada.

Flowchart proses pembelajaran LVQ tampak pada gambar di bawah ini:



Gambar 2. *Flowchart* LVQ [10]

Parameter yang dibutuhkan untuk algoritma pembelajaran LVQ adalah [5]:

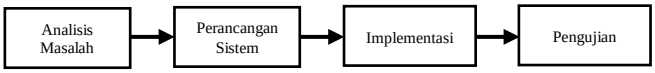
- X, vektor pelatihan ($X_1,\ldots X_i,\ldots X_n$).
- T. target untuk vektor pelatihan.
- W_j, bobot pada setiap unit vektor pelatihan ($W_{1j},\ldots W_{ij},\ldots,W_{nj}$).
- $C_{j,}$ yang merupakan kelas yang merepresentasikan unit keluaran ke-j.
- Learning rate (α), yang didefinisikan sebagai tingkat pembelajaran. Learning rate ini memiliki nilai $0 < \alpha < 1$.
- Pembaharuan bobot dilakukan dengan kondisi:
  jika $T = C_j$ maka:
  $$W_j(t+1) = w_j(t) + \alpha(t)[x(t) - w_j(t)]$$
  jika $T \neq C_j$ maka:
  $$W_j(t+1) = w_j(t) - \alpha(t)[x(t) - w_j(t)]$$

Jika menguraikan penjelasan dari *flowchart* pada gambar 2, algoritma pembelajaran dari LVQ ini adalah [11]:

1. Menginisiasi learning rate dari pembelajaran.
2. Ketika kondisi berhenti adalah false, lakukan langkah 3-7.
3. Untuk setiap input pelatihan vektor x lakukan langkah 4-6.
4. Temukan j untuk $\|x - w_j\|$ minimum.
5. Perbaharui w_j menggunakan rumus
   jika $T = C_j$ maka:
   $$W_j(t+1) = w_j(t) + \alpha(t)[x(t) - w_j(t)]$$
   jika $T \neq C_j$ maka:
   $$W_j(t+1) = w_j(t) - \alpha(t)[x(t) - w_j(t)]$$
6. Kurangi *learning rate*
7. Kondisi berhenti jika telah mencapai batas epoch atau telah mencapai kondisi yang diinginkan.

III. METODOLOGI PENELITIAN

Tahap-tahap yang dilalui dalam penelitian ini adalah sebagai berikut:



Gambar 3. Alur penelitian

A. Analisis Masalah

Tahap awal pada penelitian ini merupakan analisis masalah. Pada tahap ini masalah akan diidentifikasi dan menentukan solusi dari masalah tersebut. Hal-hal yang dianalisis berupa fungsi yang ingin dicapai oleh sistem serta masukan dan luaran sistem.

B. Perancangan Sistem

Perancangan adalah proses perencaan untuk merancang sistem yang akan dibangun. Perancangan sistem dibuat menggunakan diagram dengan tujuan memudahkan peneliti pada saat melakukan proses implementasi. Perancangan ini meliputi perancangan SAT Problem dan arsitektur LVQ.

C. Implementasi

Setelah proses perancangan selesai, kemudian masuk ke tahap implementasi. Pada tahap ini sistem yang sudah dirancang sebelumnya direalisasikan melalui proses koding.

D. Pengujian

Tahap ini bertujuan untuk melihat fungsionalitas sistem apakah sudah memenuhi standar yang diinginkan atau tidak.

IV. IMPLENTASI DAN PEMBAHASAN

A. Analisis Masalah

Fungsi yang ingin dicapai oleh sistem, yaitu sistem dapat membaca formula preposisi dengan format CNF kemudian menyelesaikan masalah *SAT Problem* menggunakan JST model LVQ.

Sistem yang ingin dikembangkan pada penelitian ini adalah sebuah pemodelan SAT Solver menggunakan JST model LVQ. *Input* dari sistem ini akan membaca formula preposisi dengan format CNF kemudian sistem akan mengeluarkan output berupa pasangan input dan output yang sudah sesuai.

Data yang digunakan dalam pelatihan maupun pengujian didapat dari situs satcompetition.com. Dari data tersebut akan dibagi menjadi data pelatihan dan pengujian.

B. Perancangan Sistem

Berikut ini akan dijelaskan rancangan SAT Problem dan Learning Vector Quantization.

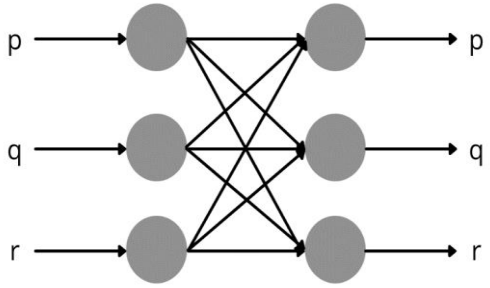
a) Rancangan LVQ

Sistem yang dibuat ini akan memberikan assignment dari setiap variable dari formula yang akan ditentuka satisfiabilitinya ketika diberikan klausa dalam formula tertentu.

Seperti yang terlihat pda gambar 1, pada arsitektur *Learning Vector Quantization*, setiap *node* menyatakan nilai setiap variabel. Oleh sebab itu banyaknya *node* pada *layer input* sama dengan banyaknya node pada variabel. Pada penelitian ini, *input* pada setiap *node* pada *layer input* dinyatakan dengan 1, 0, atau -1. Nilai 1 untuk menyatakan nilai true, 0 ketika nilai tidak diketahui, dan -1 ketika menyatakan *false*. Kemudia *output* akan menyatakan *assignment* setiap variabel yang menentukan sebuah formula tertentu SAT/UNSAT. Gambar di atas juga menunjukkan bahwa setiap banyaknya node pada *layer output* akan sama dengan banyaknya variabel. Sama seperti *input*, *node* pada setiap *layer output* akan bernilai 1, 0, atu -1 yang menyatakan *true*, tidak diketahui, atau *false*. Misalkan diketahui formula:

$$(\neg p \vee q) \wedge (p \vee r) \wedge (p \vee \neg q \vee r)$$

Formula tersebut dari 3 variabel, maka rancangan LVQ untuk problem ini seperti yang digambarkan pada Gambar 4.



Gambar 4. Rancangan model LVQ

Dari klausa di atas dapat dilihat terdapat tiga variabel, yaitu p, q, dan r yang dihubungkan dengan symbol V (OR). Serta terdapat tiga klausa yang dihubungkan dengan simbol Λ (AND).

b) Data pelatihan

Agar LVQ dapat memberikan *output* yang diharapkan, LVQ dilatih berdasarkan data pelatihan yang diperoleh dari semua klausa pada formula yang akan ditentukan satisfiabilitinya. Hal ini karena setiap klausa memuat informasi *assignment variable* yang membuat formula tersebut *satisifiable*.

Misalkan diketahui klausa (p v ¬q v r), dapat dipastikan bahwa formula tersebut akan bernilai *true* jika terdapat salah satu atau lebih dari *assignment* berikut ini:

- p bernilai *true*
- q bernilai *false*
- r bernilai *true*

Untuk membuat input jaringan syaraf tiruannya, langkah pertama yang harus dilakukan adalah mengubah formula (¬p V q) Λ (p V r) Λ (p V ¬q V r) menjadi angka 0, 1, dan -1. Angka 1 diberikan ketika variable tersebut bernilai benar, -1 ketika variabel bernilai negatif, dan 0 ketika variabel tidak diketahui. Pada klausa pertama (¬p V q), ada 2 variabel yang muncul, yaitu ¬p dan q sedangkan variable r tidak diketahui. Jadi *input* dari klausa pertama adalah -1, 1, dan 0. Kemudian cara ini diterapkan untuk setiap klausa di dalam formula matamatika yang sudah dibuat.

Dari input tersebut output yang diharapkan dari klausa (¬p V q) adalah ketika p bernilai -1 dan q bernilai 1 sehingga ada 2 kemungkinan output yang diharapkan, yaitu -1, 0, 0 dan 0, 1, 0. Detail untuk pasangan input output pada klausa di atas bisa dilihat pada gambar berikut:

p	q	r	p	q	r	
-1	1	0	-1	0	0	Klausa 1
-1	1	0	0	1	0	
1	0	1	1	0	0	Klausa 2
1	0	1	0	0	1	
1	-1	1	1	0	0	Klausa 3
1	-1	1	0	-1	0	
1	-1	1	0	0	1	

Gambar 6. Pasangan input dan output pada LVQ

Gambar di atas terlihat bahwa klausa pertama memiliki 2 pola output, klausa 2 memiliki 2 pola output, dan klausa 3 memiliki 3 pola output yang masing-masing sudah dirincikan.

Misalkan diberikan formula

$$F = C_1 \wedge C_2 \cdots \wedge C_m$$

Pada dasarnya SAT Problem ini melibatkan rumus F Boolean yang terdiri dari satu set variabel $x_1, x_2, \ldots, x_n$ dan merupakan konjungsi (gabungan) dari M klausa, $c_1, c_2, \ldots, c_m$. Klausa

merupakan disjungsi dari beberapa literal yang merupakan variable x_i atau negasinya.

C. *Implementasi*

Data pelatihan yang digunakan berbentuk format CNF dengan 16 variabel dan 18 klausa. Tahap implementasi ini dilakukan menggunakan aplikasi Python dan VS Code. Hasil dari implementasi dari rancangan sistem ini berupa assignment dari setiap varibel pada data. Library yang digunakan untuk membangun sistem adalah numpy dan pandas. *Numpy* berfungsi untuk membantu memudahkan operasi komputasi pada sistem. Pada bagian controller, dibuat modul unutk menginisiasi data yang masuk berupa file CNF dari data pelatihan. Data yang dimasukkan ke dalam sistem berbentuk .csv oleh karena itu diperlukan *library pandas* untuk bisa membaca *file* tersebut. Kemudian modul tersebut akan dipross pada bagian modul SAT untuk bisa diuraikan menjadi angka 1, 0, dan -1 sehingga menjadi seperti gambar dibawah ini:

```
[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, -1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, -1]
[0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, -1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1]
```

Gambar 7. Data *input*

Gambar ini menunjukkan input dari data pelatihan yang digunakan. Input ini dibuat pasangan outputnya sesuai dengan rangan LVQ yang telah dibuat sebelumnya sehingga menghasilkan output seperti gambar di bawah ini:

```
[[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
[[0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
[[0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
[[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, (
[[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, (
[[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0
[[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1
[[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0
[[0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, (
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, -1, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1
```

Gambar 8. Data *output*

Gambar 7 dan 8 merupakan pasangan input output dari data pelatihan yang digunakan. Untuk bisa mendapatkan output seperti gambar di atas, diperlukan proses pembelajaran dengan metode LVQ. Data yang sudah diuraikan menjadi pasangan *input* dan *output* tersebut kemudia diproses pada bagian LVQ untuk dilakukan pelatihan sehingga bisa masuk ke bagian pengujian.

D. *Pengujian*

Setelah pelatihan selesai, pengujain dilakukan dengan salah satu atau beberapa klausa. Kemudian dicek apakah output tersebut akan membuat formula bernilai *true*/*false*. Tahap pengujian ini dilakukan untuk melihat sistem mampu menyelesaikan *SAT Problem* setelah melakukan proses pelatihan. Output yang keluar merupakan data yang memiliki jarak terdekat dengan semua data pelatihan. Contohnya untuk klausa (¬p v q) yang kemudian diubah menjadi angka -1, 1, dan 0. Kemudian nilai dari *input* tersebut akan bernilai:

- p bernilai *false*
- q bernilai *true*
- r tidak diketahui

Dari data tersebut, output yang diharapkan adalah yang memiliki pola -1, 0, 0, dan 0, 1, 0.

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1]

[[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]]
```

Gamabar 9. Pengujian pasangan input output

Gambar di atas merupakan hasil pengujian untuk pasangan input dan output yang telah dilakukan dengan menggunakan salah satu data pelatihan. Hasilnya adalah sistem bisa memberikan output yang memiliki jarak terdekat dengan data peatihan. Selanjutnya untuk melihat data tersebut bernilai satisfiable atau tidak, dilakukan pengujian dengan menggunakan satu formula secara utuh. Jika dalam pengujian ini lebih banyak mengeluarkan data yang memiliki nilai sesuai dengan pasangan inputnya, maka formula tersebut bernilai satisfiable. Tetapi jika keluarannya lebih banyak yang tidak sama dengan nilai outputnya maka formula tersebut bernilai unsatisfiable. Hasil dari pengujian tersebut bisa dilihat pada gambar 10.

```
Result:
[[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0]] << different
[[0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0]] << different
[[0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1]]
[[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]]
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]]
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]
```

Gambar 10. Hasil pengujian

```
Total Data:  18
Total Equal Data:  16
This data is SAT
```

Gambar 11. Hasil pengujian data

Gambar di atas menampilkan *output* sesuai dengan *input* yang diuji. Dari gambar 10 bisa dilihat ada 2 data yang tidak sesuai dengan *input* dan sisanya bernilai sama. Oleh sebab itu sistem akan mengeluarkan hasil seperti gambar 11 dan bisa menyatakan bahwa formula yang diujikan bernilai *satisfiable*.

V. KESIMPULAN DAN SARAN

Kesimpulan dari penelitian ini adalah telah berhasil dibentuk sistem yang mampu untuk menyelesaikan *SAT*

Problem yang dibangun menggunakan bahasa pemograman *Python*. Pada tahap pengujian, sistem sudah bisa menampilkan *output* yang jaraknya paling dengan data pelatihan dan bisa menampilkan formula bernilai *satisfiable*.

Saran untuk kedepannya agar penelitian ini bisa dikembangkan lebih lanjut menggunakan data pelatihan yang lebih besar dan kompleks agar dapat dilhat sejauh mana keberhasilan *SAT Solver* dengan menggunakan LVQ.

REFERENCES

[1] A. Bhattacharjee and P. Chauhan, "Solving the SAT problem using genetic algorithm," *Adv. Sci. Technol. Eng. Syst.*, vol. 2, no. 4, pp. 115–120, 2017.

[2] A. S. Ritonga and S. Atmojo, "Pengembangan Model Jaringan Syaraf Tiruan untuk Memprediksi Jumlah Mahasiswa Baru di PTS Surabaya (Studi Kasus Universitas Wijaya Putra)," *J. Ilm. Teknol. Inf. Asia*, vol. 12, no. 1, p. 15, 2018.

[3] B. Bünz and M. Lamm, "Graph Neural Networks and Boolean Satisfiability," pp. 1–9, 2017.

[4] D. Selsam, "Neural Networks and the Satisfiability Problem a Dissertation Submitted To the Department of Computer Science and the Committee on Graduate Studies of Stanford University in Partial Fulfillment of the Requirements for the Degree of," no. June, 2019.

[5] E. Budianita and W. Prijodiprodjo, "Penerapan Learning Vector Quantization (LVQ) untuk Klasifikasi Status Gizi Anak," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 7, no. 2, p. 155, 2013.

[6] G. Afan and L. Yen, "Implementasi Jaringan Syaraf Tiruan Recurrent," *J. ComTech*, vol. 02, no. 01, pp. 197–208, 2010.

[7] H. T. Frianto and M. Rivai, "Implementasi Jaringan Syaraf Tiruan Backpropagation Dan Self Organizing Map Menggunakan Sensor Gas Semikonduktor Sebagai Identifikasi Jenis Gas," *Epoch*, vol. 2008, no. semnasIF, pp. 219–228, 2008.

[8] I. Bayu Priyanta and I. Astawa, "Penerapan Jaringan Syaraf Tiruan Dalam Prakiraan Hujan Harian Di Daerah Kuta Selatan Provinsi Bali," *J. Ilmu Komput.*, vol. 7, no. 1, pp. 7–11, 2014.

[9] M. K. R. G.-B. P. J. V.-M. J.-M. A. 1Chaki b Boualloub, "Universitas Islam Negeri Maulana Malik Ibrahim," *Peran Sungai bagi Masy.*, vol. 9, no. 1, pp. 76–99, 2010.

[10] S. R. Dilfa Yustia Quar'aini, "JARINGAN SYARAF TIRUAN LEARNING VECTOR QUANTIZATION UNTUK APLIKASI PENGENALAN TANDA TANGAN 1 Difla Yustisia Qur'ani 1 , Safrina Rosmalinda 2," *Snatika*, vol. 2010, no. Snati, pp. 1–5, 2010.

[11] R. Meliawati, O. Soesanto, and D. Kartini, "Penerapan Metode Learning Vector Quantization (LVQ) Pada Prediksi Jurusan Di SMA PGRI 1 Banjarbaru," *Kumpul. J. Ilmu Komput.*, vol. 04, no. 01, pp. 11–20, 2016.

[12] T. Hidayat, "SAT Solver dengan DPLL dalam Pemograman Deklaratif," *Semin. Nas. Apl. Teknol. Inf.*, 2018.

[13] Y. Vizel, G. Weissenbacher, and S. Malik, "Boolean Satisfiability Solvers and Their Applications in Model Checking," *Proc. IEEE*, vol. 103, no. 11, pp. 2021–2035, 2015.

[14] T. H. Alexander Ramadhan Suratinoyo, "Penentuan Solusi Satisfiability (SAT) Problem Dengan Metode Kohonen Self-Organizing Map (K- SOM)," *jURIKOM*, vol. 3, no. 2, pp. 101–120, 2020.

[15] Pratama, "Penyelesaian Boolean Satisfiability Problem Dengan Algoritma Davis Putnam Logemann Loveland (Dpll) Menggunakan Java Dengan Algoritma Davis Putnam Logemann Loveland (Dpll) Menggunakan Java," 2018.

[16] M. A. Setiawan, "Solusi Pendekatan SAT Problem Dengan Algoritma Genetika Menggunakan Java," 2019.