mgr inż. WITOLD GŁOWACZ
Akademia Górniczo-Hutnicza im. Stanisława Staszica

Koncepcja sterowania maszyny elektrycznej z zastosowaniem automatycznego systemu dialogowego

W artykule przedstawiono koncepcję sterowania maszyny elektrycznej z zastosowaniem automatycznego systemu dialogowego opartego na języku Python i utworzonej w tym języku bazie wiedzy. System dialogowy pozwala nawiązać kontakt w języku naturalnym pomiędzy użytkownikiem i botem. Bot analizuje zdania użytkownika i generuje swoje zdania. Język naturalny jest zdefiniowany za pomocą metod formalnych, w tym gramatyk precedensyjnych i automatów skończonych. Opracowano w języku Python program, który pozwala prowadzić rozmowę pomiędzy użytkownikiem i botem. W końcowym etapie kod programu w języku Python jest przekształcany do języka wewnętrznego mikrokontrolera za pomocą modułu PyMite. System dialogowy może być użyty do sterowania maszyn elektrycznych w górnictwie i zwiększenia poziomu bezpieczeństwa.

1. WSTĘP

Systemami dialogowymi nazywamy systemy informatyczne, które pozwalają prowadzić rozmowę z użytkownikami. Składają się one z bota (softwarowego robota) i bazy wiedzy. Rozmowa użytkownika z botem może być prowadzona w różnych językach, w szczególności w naturalnym języku polskim. Bot analizuje zdania użytkownika i formułuje własne zdania, używając reguł wybranego języka i bazy wiedzy [3, 5]. Nowoczesne systemy dialogowe mają własności systemów ekspertowych i mogą być zastosowane do sterowania maszyn elektrycznych. Systemy takie przechowują zdobytą wiedzę uzyskaną z treningu i doświadczenia. Jednym z nich jest język Python, napisany przez Guido van Rossuma w 1990 roku [8]. W dostępnej literaturze można znaleźć porównania Pythona do takich języków, jak Perl, PHP, C/C++ czy Java. Python jest w równym stopniu podobny do każdego z wymienionych, jak i różny od nich. Podobieństwa i różnice łatwo określić, zapoznając się z kilkoma jego podstawowymi cechami:

- Python jest językiem w pełni obiektowym. Odróżnia go to zarówno od Perla, jak i od PHP (pomimo faktu, że PHP z wersji na wersję coraz bardziej wspiera obiektowość – wciąż jednak czyni to słabo).
- Pomimo tego, że jest językiem obiektowym, Python nie wymusza obiektowego stylu programowania. Pozwala programiście tworzyć również programy strukturalne i funkcyjne.
- Zmienne nie mają typów, wartości. Ściślej rzecz ujmując, typy zmiennych w Pythonie są w pełni dynamiczne. Oznacza to podobną do PHP łatwość zmiany zastosowania zmiennych, choć jednocześnie takie rozwiązanie potrafi dawać inne efekty niż w Javie czy C++, gdzie przypisywane wartości muszą być zgodne z typem zmiennej i kompilator ściśle tego pilnuje.
- Python jest językiem interpretowanym – podobnie jak PHP czy Perl, a w przeciwieństwie do C/C++ czy Javy. To znacznie ułatwia tworzenie i modyfikowanie programów, jednak nieco negatywnie odbija się na wydajności. Sytuację poprawia wbudowany mechanizm konwersji kodów źródłowych do zrozumiałej dla interpretera postaci binarnej, co przy kolejnych uruchomieniach skryptu może dawać spory wzrost wydajności.

- Python, podobnie jak Java, jest dostarczany wraz z bogatym zestawem bibliotek (głównie dedykowanych operacjom sieciowym). Python szeroko korzysta z dziedziczenia, dlatego łatwo jest nie tylko implementować biblioteki w programach, ale także je dziedziczyć, rozszerzać i tworzyć nowe. Dzięki temu, podobnie jak w przypadku Javy, powstaje wiele ciekawych rozszerzeń tych bibliotek, tworzonych przez niezależnych programistów.
- Środowisko Pythona jest w pełni interaktywne. Można na bieżąco wprowadzać kolejne polecenia i oglądać wyniki. Bardzo ułatwia to tworzenie i debugowanie programów pisanych w tym języku.
- Python intensywnie się rozwija, z tendencją do wprowadzania maksymalnej zwięzłości kodu. Przybliża go to do Perla, co nie podoba się części webmasterów, którzy twierdzą, że dzieje się to ze szkodą dla przejrzystości kodu.

2. STEROWANIE GŁOSEM I TEKSTEM

Programy rozpoznawania mowy występują w dwóch wariantach [9]:

1. Mały słownik, wielu użytkowników. Jest to system idealny dla automatycznych biur obsługi. Użytkownik może mówić w niemal dowolny sposób – z akcentem polskim, niemieckim, południowokoreańskim, jąkając się i ciężko dysząc, a system i tak zrozumie większość wypowiadanych kwestii. Problem polega na tym, że liczba komend jest niewielka i dotyczy podstawowych funkcji menu, np. komend typu „usuń” lub „dalej” oraz liczb.
2. Duży słownik, mniej użytkowników. Tego typu system najlepiej funkcjonuje w środowisku, gdzie z programem będzie mieć do czynienia niewielka liczba użytkowników. Co prawda, jego dokładność jest relatywnie duża, bowiem sięga przynajmniej 85% w przypadku doświadczonych użytkowników, a zasób jego biblioteki liczy się w dziesiątkach tysięcy słów, ale skutkuje najlepiej w przypadku wąskiej grupy, przyzwyczajonej do komunikacji z programem. Im bardziej grupa się rozszerza i staje się przypadkowa, tym bardziej skuteczność spada.

Osobna kwestia dotyczy wyboru pomiędzy mówieniem ciągłym a takim, gdzie słowa są od siebie wyraźnie oddzielone. Oczywiście systemowi jest dużo łatwiej zrozumieć wyrazy, kiedy wymawiamy je czysto, dodatkowo robiąc między nimi efektowne pauzy. Użytkownicy wolą jednak mówić w naturalnym, konwersacyjnym tempie, choć najczęściej – dla bezpieczeństwa – preferują niemalże sylabizowanie. Gdyby mogli mówienie poprzeć gestykulacją, na pewno skorzystaliby z tej dodatkowej pomocy. Na szczęście większość stosowanych programów jest w stanie zrozumieć mowę ciągłą.

Aby zamienić mowę na tekst lub komendy, system musi dokonać kilku złożonych czynności. Kiedy mówimy, tworzymy w powietrzu wibracje. Następnie analog-to-digital converter (ADC, czyli konwerter analogowo-cyfrowy) przekłada analogową falę na cyfrowe dane, które komputer jest w stanie zrozumieć. Aby tego dokonać, pobiera próbki dźwięku przez dokładne pomiary fali w krótkich odstępach czasu. System filtruje zdigitalizowany dźwięk, aby pozbyć się szumów i zakłóceń, oraz rozdziela go na różne pasma częstotliwości. Poza tym dostosowuje dźwięk do stałego poziomu głośności i go wyrównuje (trzeba pamiętać, że ludzie zazwyczaj nie mówią w tym samym tempie, tak więc dźwięk musi zostać dostosowany do szablonów, znajdujących się w pamięci systemu).

Kolejną operacją jest dzielenie sygnału na niewielkie części, długości setnych lub nawet tysięcznych części sekundy, w przypadku zbitek spółgłoskowych (czyli grupy występujących po sobie spółgłosek o zbliżonym brzmieniu, np. **roz-pierz**-chłszy, u-**sz-cz**-knąć lub naj-**wstrze-mię**-źliwiej). Następnie program dopasowuje te części do znanych fonemów w odpowiednim języku (fonemy to najmniejsze części mowy, których w języku polskim jest – w zależności od przyjętych założeń – między 36 a 44, w angielskim – około 40, natomiast w jednym z języków występującym na terenie Afryki Południowej – aż 112).

Dalszy krok wydaje się na pozór prosty, ale jest w gruncie rzeczy najtrudniejszy do realizacji i stanowi kluczowy element wielu badań związanych z rozpoznawaniem mowy. Chodzi o to, aby program przebadał fonemy w kontekście innych fonemów w ich otoczeniu. Analiza jest dokonywana przy zastosowaniu złożonego modelu statystycznego, porównującego fonemy do biblioteki znanych słów, wyrażeń i zdań. W ten sposób program jest w stanie określić, co prawdopodobnie mówi użytkownik, i zamienić fonemy na pisany tekst, komendę lub zapytanie.

Programy, które rozpoznają tekst, występują również w dwóch wariantach: mały słownik i duży słownik [4]. W sterowaniu maszyn elektrycznych zastosowanie mają programy wariantu „mały słownik”. Języki użytkownika i bota można zdefiniować za pomocą gramatyk precedensyjnych [1, 2] lub za pomocą deterministycznych automatów skończonych [2].

Gramatyką precedensyjną nazywamy uporządkowaną czwórkę $\mathbf{G} = < \mathbf{N}, \mathbf{\Sigma}, \mathbf{P}, S >$, gdzie:
$\mathbf{N}$ – skończony zbiór symboli nieterminalnych,
$\mathbf{\Sigma}$ – skończony zbiór symboli terminalnych,
$\mathbf{P} \subset \mathbf{N} \times (\mathbf{N} \cup \mathbf{\Sigma})^{*}$ – skończony zbiór produkcji,
$S \in \mathbf{N}$ – głowa alfabetu.

Deterministycznym automatem skończonym (DFA) nazywamy uporządkowaną piątkę $A=<Q,\Sigma,\delta,q_{start},F>$,

gdzie:

Q – skończony zbiór stanów,

Σ – alfabet,

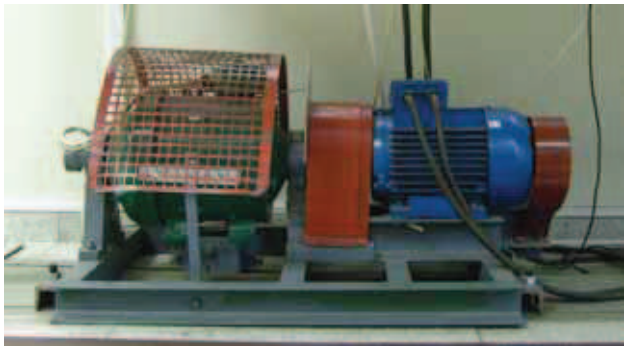
$\delta: Q\times\Sigma\rightarrow Q$ – funkcja przejścia ($\delta(q,a)$ – stan, do którego przechodzi automat ze stanu q po przeczytaniu litery a),

$q_{start}\in Q$ – stan początkowy automatu,

$F\subset Q$ – zbiór stanów akceptujących (automat, przechodząc do tego stanu, akceptuje dotychczas przeczytane słowo).

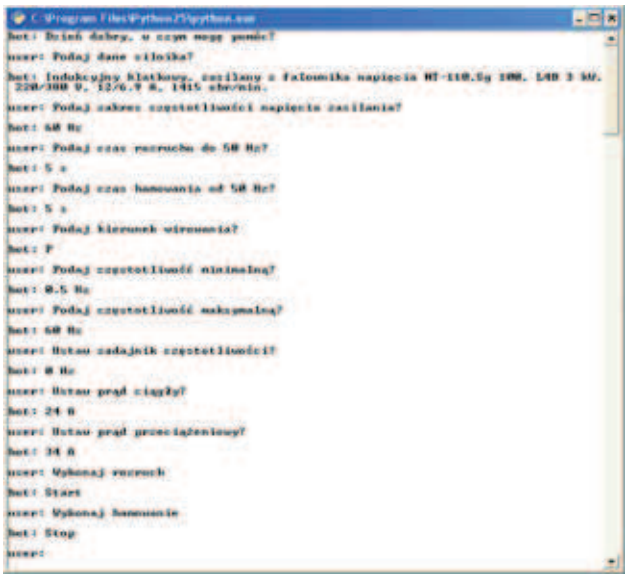
3. PRZYKŁADOWY DIALOG STEROWANIA

W języku Python opracowano program, który pozwala prowadzić rozmowę pomiędzy użytkownikiem i botem. Do badań wybrano silnik indukcyjny klatkowy (rys. 1).



Rys. 1. Badany silnik indukcyjny klatkowy [3]

Bot prowadzi z użytkownikiem następującą rozmowę (rys. 2):



Rys. 2. Fragment rozmowy użytkownika z botem, prowadzonej za pomocą aplikacji napisanej w języku Python [3]

4. PRZEKSZTAŁCANIE KODU PROGRAMU W JĘZYKU PYTHON DO JĘZYKA WEWNĘTRZNEGO MIKROKONTROLERA

Kod programu w języku Python jest przekształcany do języka wewnętrznego mikrokontrolera za pomocą modułu PyMite [6]. Innym rozwiązaniem jest translacja programu w języku Python na język C i przekształcenie go do języka wewnętrznego mikrokontrolera w środowisku Atmel Studio 6 [7].

Program po komendzie „Start” wykonuje rozruch silnika indukcyjnego klatkowego, a po komendzie „Stop” – hamowanie silnika zgodnie z danymi wymienionymi w dialogu.

5. PODSUMOWANIE

Metody formalne pozwalają zdefiniować języki użytkownika i bota. Użycie języka naturalnego jest trudne ze względu na rozbudowaną strukturę. W języku Python opracowano program, który pozwala prowadzić rozmowę pomiędzy użytkownikiem i botem. Z przeprowadzonych badań wynika, że zastosowanie bazy wiedzy w rozpoznawaniu poszczególnych słów użytkownika i identyfikacji zdań użytkownika jest skuteczną techniką rozpoznawania informacji. Metody przetwarzania języka naturalnego mogą być wykorzystane w projektowaniu automatycznych systemów dialogowych. Systemy dialogowe ze względu na dużą szybkość działania i posiadaną wiedzę mogą być stosowane do sterowania maszyn elektrycznych.

Praca została sfinansowana ze środków statutowych AGH, umowa AGH nr 11.11.120.815 (Witold Głowacz).

Literatura

1. Aho A.V., Ullman J.D.: *The Theory of Parsing, Translation and Compiling*, Vol. I: Parsing, Vol. II: Compiling, Englewood Cliffs, Prentice-Hall, 1973.
2. Blikle A.: *Automaty i gramatyki*. Wstęp do lingwistyki matematycznej, Warszawa, PWN, 1971.
3. Głowacz W.: *Implementacja automatycznego systemu dialogowego oparta o język Python i bazę danych MySQL*. „Mechanizacja i Automatyzacja Górnictwa", nr 10, 2011, Katowice, s. 11-14.
4. Lubaszewski W.: *Słowniki komputerowe i automatyczna ekstrakcja informacji z tekstu*, Uczelniane Wydawnictwa Naukowo-Dydaktyczne AGH, 2009, s. 257-260.
5. Santangelo A., Augello A., Gentile A., Pilato G., Caglio S.: *A Chat-bot based Multimodal Virtual Guide for Cultural Heritage Tours. Proc. of PSC*, Las Vegas, 2006, pp. 114-120.
6. http://www.slideshare.net/stoggi/python-on-a-chip.
7. http://www.atmel.com.
8. http://internetmaker.pl/artykul/3423,1,wprowadzenie_do_pythona_-_jezyk_oprogramowania_inny_niz_wszystkie.
9. http://pej.cz/Siri-i-kumple-jak-dziala-rozpoznawanie-mowy-a1265.

Artykuł został zrecenzowany przez dwóch niezależnych recenzentów.

Z ŻYCIA EMAG-u

EMAG NA TARGACH AMPER 2013

Instytut EMAG wziął udział w Targach AMPER 2013, które w dniach **19-22 marca 2013 r.** odbyły się w czeskim Brnie.

Międzynarodowe Targi Elektroniki, Elektrotechniki, Automatyki i Technik Komunikacyjnych AMPER zaliczają się do najważniejszych imprez targowych w branży elektrotechnicznej i elektronicznej w regionie Europy Środkowej i Wschodniej. W tegorocznej, 21. edycji imprezy wystawcy z Polski wzięli udział w ramach Wyjazdowej Misji Gospodarczej, zorganizowanej przez Stowarzyszenie Elektryków Polskich. Podczas imprezy Instytut EMAG zaprezentował ofertę Centrum Badań i Certyfikacji w zakresie certyfikacji wyrobów oraz badań laboratoryjnych, w tym m.in. nowe możliwości badawcze Laboratorium Badań Kompatybilności Elektromagnetycznej w zakresie urządzeń dla branży automotive oraz Laboratorium Badań Kabli i Badań Środowiskowych z zakresu badań klimatycznych i wibracyjnych.

SUKCES W SANKT PETERSBURGU

Przedstawiciele Instytutu EMAG wzięli udział w międzynarodowej konferencji naukowej dla młodych naukowców poświęconej problemom eksploatacji minerałów, która odbyła się w Sankt Petersburgu w dniach **24-26 kwietnia 2013 r**.

8. Międzynarodowa Konferencja „Problemy eksploatacji minerałów" zorganizowana została w Narodowym Mineralno-Surowcowym Uniwersytecie „Górniczym" w Sankt-Petersburgu, jednej z najstarszych i największych uczelni górniczych w Europie. W tegorocznej edycji tego wydarzenia wzięli udział uczestnicy z ponad 70 uczelni oraz jednostek naukowych z 21 krajów, w tym również przedstawiciele Instytutu Technik Innowacyjnych EMAG. W towarzyszącym forum konkursie promującym najlepszych doktorantów i młodych naukowców pracownik Instytutu EMAG, mgr inż. Jarosław Smyła, zajął drugie miejsce w kategorii „Geologia", wygłaszając referat pt. „Ocena zawartości popiołu w węglu przez pomiar jego naturalnej promieniotwórczości gamma".