

# 연결 단어 음성 인식기 학습용 음성DB 녹음을 위한 최적의 대본 작성 알고리즘

## The Optimal and Complete Prompts Lists Generation Algorithm for Connected Spoken Word Speech Corpus

유 하 진\*  
(Ha-Jin Yu\*)

\*서울시립대학교 컴퓨터과학부

(접수일자: 2003년 11월 27일; 채택일자: 2004년 2월 3일)

연결 단어 인식기, 특히 연결 숫자음 인식기를 제작하기 위한 음성 데이터베이스를 구축하는데 있어서 완전하고 효율적인 발성목록을 작성하기 위한 알고리즘을 제안한다. 기존의 음성 DB에서 사용되는 목록은 주로 난수 발생기에 의하여 만들어지거나 사용자의 전화번호, 우편번호 등을 이용하여 만들어져 왔으므로 다양한 환경의 음소 또는 단어를 균일하게 포함하고 있지 못하다. 따라서 본 논문에서는 하나의 단어에 대하여 전후에 모든 단어가 연결되는 조합을 모두 한번씩 포함하는 목록을 만드는 효율적인 알고리즘을 제안한다. 본 알고리즘으로 7연 숫자 목록을 만들면 200개의 문장으로 모든 조합을 포함할 수 있게 된다. 본 논문에서는 알고리즘 예제와 본 알고리즘의 완전성과 효율성에 대하여 기술하였다.

**핵심용어:** 음성인식, 한국어, 연결 단어 인식, 연결 숫자음 인식, 음성 DB

**투고분야:** 음성처리 분야 (2.5)

This paper describes an efficient algorithm to generate compact and complete prompts lists for connected spoken words speech corpus. In building a connected spoken digit recognizer, we have to acquire speech data in various contexts. However, in many speech databases the lists are made by using random generators. We provide an efficient algorithm that can generate compact and complete lists of digits in various contexts. This paper includes the proof of optimality and completeness of the algorithm.

**Keywords:** Speech recognition, Korean, Connected word recognition, Connected digit recognition, Speech corpus

**ASK subject classification:** Speech signal processing (2.5)

### I. 서론

음성인식에 있어서 숫자음 인식은 가장 기본적인 일이라고 할 수 있다. 숫자는 일상에서 가장 많이 쓰이는 어휘이고, 그 어휘수가 10개 정도로 매우 작아서 쉽게 인식기를 만들 수 있다. 완벽한 숫자인식기가 완성되면 자동 다이얼링, 메뉴 선택 등 그 규모에 비하면 용도는 이루 열거할 수 없을 정도로 많을 것이다. 그런데, 영어 등 외국어의 경우에는 숫자음을 인식하는 데 큰 어려움이 없으나, 한국어의 경우에는 숫자음 인식이 음성인식 과제 중에서 가장 어려운 종류에 속한다. 그 이유는 한국어

숫자음이 모두 한 음절 또는 한 음소로 이루어져 있기 때문이다. 한국어 숫자음 인식의 난이도는 음절인식이나 음소인식과 거의 비슷하다고 할 수 있다. 이러한 경우는 단어에 포함된 음향학적 정보가 적어서 단어를 서로 구분하기 어렵게 된다. 이와 같은 이유로 한국어 숫자음 인식은 이미 실용화 단계에 와 있는 영어 숫자음 인식과는 달리 아직도 큰 과제로 남아있다. 따라서, 한국어 연결 숫자음 인식은 연결 음소인식과 같은 방법으로 해결해야 할 필요가 있다. 연속 음성에서의 음소 인식에서 좌우 문맥을 고려하는 트라이폰을 사용하여 높은 성능을 얻을 수 있듯이 숫자음 인식에서도 숫자음의 좌우 문맥을 고려한 모델링을 하면 보다 높은 성능을 얻을 수 있을 것이다.

이와 같이 연결 숫자음 음성인식기를 만들 때 가장 먼저 해야 할 일은 많은 사람이 발성한 음성자료를 수집하

책임저자: 유하진 (hiju@venus.uos.ac.kr)  
130-743 서울시 동대문구 전농3동  
서울시립대학교 컴퓨터과학부  
(전화: 02-2201-5618; 팩스: 02-2210-5274)

는 일이다. 이를 위해서는 다양한 음소환경에서 숫자음이 발생되도록 발생 목록을 만드는 것이 필요하다. 이 발생 목록은 또한 그 양이 너무 많지 않아 소수의 인원이 전체를 발생할 수 있도록 해야 한다. 그런데, 현재까지 세계 각국에서 수집된 많은 연결 숫자음 음성 DB에서 사용된 발생 목록은 최적의 목록이라고 보기 어렵다[1][2]. 예를 들어, 대부분의 목록은 난수발생기를 사용하여 만들어져 있거나[3], 인접한 단어간의 분포만을 고려하고 있다[4]. OGI의 3만 숫자음 음성코퍼스[5]에서는 발생한 사람의 우편번호나 번지수, 전화번호 등 임의의 번호를 사용하고 있다.

좌우 문맥이 고려된 숫자음 인식기를 만들기 위해서는 좌우 문맥을 고려하여 녹음된 음성DB가 필요하게 된다. 즉, 모든 숫자의 좌우에 다른 숫자가 이어지는 모든 조합을 고려하는 것이다. 10개의 숫자를 사용한다고 할 때 좌우 문맥이 반영된 숫자의 개수는  $10^3 = 1,000$ 개가 된다. 그런데, 0을 공 또는 영 두가지로 발생하고, 6을 육, 륙 등 두 가지로 발생하는 것을 허용한다고 하면 문맥종속 숫자음의 개수는  $12^3 = 1,728$  개로 급격히 증가하게 된다. 그러나 한 문장에 여러 개의 문맥종속 숫자음이 포함되고, 전체 문장 집합에 모든 문맥종속 숫자음이 중복되지 않게 포함되게 할 수 있다면 전체 문장의 수를 충분히 작게 할 수 있다. 문장의 처음과 끝에서의 문맥을 고려하지 않을 때, 하나의 문장이  $n$ 개의 숫자음을 포함한다면 하나의 문장에  $n-2$ 개의 좌우문맥종속 숫자음을 포함시킬 수 있다. 예를 들어 10개의 단어를 사용하고, 7연숫자를 사용한다면 하나의 발생 당 5개의 좌우 문맥을 포함할 수 있으므로, 200개의 문장으로 1,000개의 좌우문맥종속 숫자음을 모두 표현할 수 있다. 이것은 한 사람의 화자도 충분히 발생할 수 있는 양이다. 이때, 어휘는 반드시 숫자음만으로 제한되는 것이 아니고, 어느 어휘에든지 적용할 수 있다.

본 논문에서는 이와 같은 최적의 발생목록을 만들어 내는 알고리즘을 제안한다. 2장에서는 알고리즘을 기술하고 3장에서는 간단한 예를 들어 설명하며, 4절에서는 제안한 알고리즘의 완전성과 효율성을 검증한다.

## II. 제안한 목록 생성 알고리즘

어휘 집합  $V$ 에  $T$ 개의 단어가 있다고 가정하고, 하나의 발생용 문장에  $n$ 개의 단어를 넣는다고 하자. 즉,  $n$ 연 숫자

음 목록을 만든다고 하자. 여기서 편의상  $T/(n-2)$ 이 정수라고 가정하자. 그렇지 않은 경우에 대해서는 4장에서 언급한다. 어휘 집합  $V$ 는 다음과 같이 숫자로 표현할 수 있다.

$$V = \{0, 1, \dots, T-1\}$$

이때, 1차 및 2차 차분 수열을 다음과 같이 정의하도록 한다.

**정의 1 (1차 차분 수열):**

최종 수열 집합의 한 문장을

$$S = (s_0, s_1, \dots, s_{n-1}), s_i \in V$$

이라고 하면, 이 문장의 1차 차분수열을 다음과 같이 정의한다.

$$D = (d_0, d_1, \dots, d_{n-2}),$$

$$s_{i+1} = (s_i + d_i) \bmod T \tag{1}$$

즉, 1차 차분수열은 수열  $S$ 내의 이웃한 숫자들 간의 차이로 이루어져 있는 수열이다.

**정의 2 (2차 차분 수열):**

2차 차분수열은 1차 차분수열  $D$ 내의 이웃한 숫자들 간의 차이로 이루어져 있는 수열로 다음과 같이 정의한다.

$$A = (a_0, a_1, \dots, a_{n-3})$$

$$d_{i+1} = (d_i + a_i) \bmod T \tag{2}$$

그림 1은 정의를 설명하고, 그림 2는  $T=10$  일 때의 예를 보여준다. 좌우 종속 단어열을 생성하는 알고리즘은 다음과 같다.

**알고리즘:** 좌우문맥종속 단어열 생성

입력 :

$T =$  어휘 집합의 단어 수

$n =$  한 문장내의 단어 수.  $T/(n-2)$ 는 정수라고 가정

출력:

$T^3/(n-2)$  개의 좌우문맥종속 숫자열

$$S_i = (s_{i,0}, s_{i,1}, \dots, s_{i,n-1})$$

여기서  $s_{i,j}$ 는  $i$ 번째 숫자열의  $j$ 번째 수를 말한다.

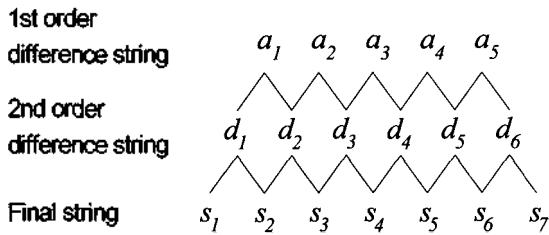


그림 1. 1차, 2차 차분수열의 정의  
Fig. 1. The definitions of the first and second difference sequences.

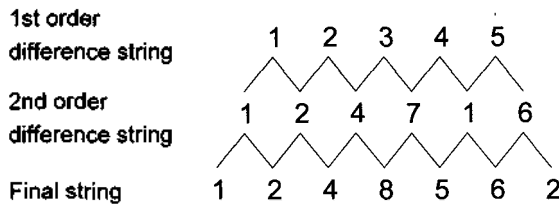


그림 2. 수열의 예  
Fig. 2. An example of the digit sequences.

- $T/(n-2)$ 개의 서로 다른 2차 차분수열  $A_k = (a_{k,0}, a_{k,1}, \dots, a_{k,n-3}), k = 0, 1, 2, \dots, T/(n-2)-1, a_{k,i} \in V$ 을 임의로 생성한다. 각각의 숫자  $a_{k,i}$ 의 순서는 중요하지 않으나, 모든 숫자가 반드시 **한번씩** 사용되어야 한다.
- 정의 2에 의하여  $T/(n-2)$ 개 각각의 2차 차분수열  $A_k$ 에 대하여  $T$ 개의 서로 다른 1차 차분수열  $D_j = (d_{j,0}, d_{j,1}, \dots, d_{j,n-2}), j = 0, 1, 2, \dots, T-1$ 을 생성한다. 여기서  $d_{j,0} = j$ 가 되도록 한다.
- 정의 1에 의하여  $T$ 개 각각의 1차 차분수열에 대하여  $T$ 개의 최종 수열  $S = (s_{i,0}, s_{i,1}, \dots, s_{i,n-1}), s_{i,0} = i, i = 0, 1, \dots, T-1$ 를 생성한다.

### III. 문장 생성의 예

본 장에서는 제안한 알고리즘의 실행 예를 보여준다. 어휘수가 4라고 가정하고  $n$ 을 6으로 잡으면  $T/(n-2) (= 4/(6-2) = 1)$ 가 정수가 된다. 그러면 최종 6연숫자의 수는  $T^3/(n-2) = 4^3/(6-2) = 16$ 개가 된다.

먼저 알고리즘의 단계 1에서와 같이 2차 차분수열을 다음과 같이 만들 수 있다.

$$A_0 = (0, 1, 2, 3)$$

다음은 알고리즘의 단계 2에 의하여 4개의 1차 차분수

열을 만든다.

$$D_0 = (0, 0, 1, 3, 2)$$

$$D_1 = (1, 1, 2, 0, 3)$$

$$D_2 = (2, 2, 3, 1, 0)$$

$$D_3 = (3, 3, 0, 2, 1)$$

여기서 각 수열의 초기값에는 0,1,2,3이 각각 사용되었다. 이 수열 내에는 인접한 두 수의 조합  $(d_{j,b}, d_{j,i+1})$ 에 중복되는 것이 없음을 알 수 있다. 마지막으로 다음과 같이 16개의 최종 6연 숫자열을 만들 수 있다. 이 수열들을 조사해 보면 세 개의 연속된 조합이 중복된 것이 없음을 알 수 있다. 또한, 모든 2개의 연속된 숫자의 조합은 같은 수만큼 존재함을 알 수 있다. 예를 들어 연속된 두 수의 조합 (2,3)은 정확히 5개가 출현한다.

000102	012003	020300	032201
111213	123110	131011	103312
222320	230221	202122	210023
333031	301332	313233	321130

10개의 숫자를 사용하여, 7연 숫자 열을 만들기 위해서는 다음과 같은 두개의 2차 차분수열을 이용하면 된다.

$$A_0 = (0, 1, 2, 3, 4)$$

$$A_1 = (5, 6, 7, 8, 9).$$

이를 이용하여 만들어진 다음 200개의 7연 숫자 열은 가능한 좌우문맥종속 숫자를 모두 포함하고 있다.

0014005	0564050	0137451	0687406	0250807
0700852	0373253	0823208	0496609	0946654
0519055	0069000	0632401	0182456	0755857
0205802	0878203	0328258	0991659	0441604
1125116	1675161	1248562	1798517	1361918
1811963	1484364	1934319	1507710	1057765
1620166	1170111	1743512	1293567	1866968
1316913	1989314	1439369	1002760	1552715
2236227	2786272	2359673	2809628	2472029
2922074	2595475	2045420	2618821	2168876
2731277	2281222	2854623	2304678	2977079
2427024	2090425	2540470	2113871	2663826
3347338	3897383	3460784	3910739	3583130
3033185	3606586	3156531	3729932	3279987
3842388	3392333	3965734	3415789	3088180
3538135	3101536	3651581	3224982	3774937

4458449 4908494 4571895 4021840 4694241  
 4144296 4717697 4267642 4830043 4380098  
 4953499 4403444 4076845 4526890 4199291  
 4649246 4212647 4762692 4335093 4885048  
 5569550 5019505 5682906 5132951 5705352  
 5255307 5828708 5378753 5941154 5491109  
 5064500 5514555 5187956 5637901 5200302  
 5750357 5323758 5873703 5446104 5996159  
 6670661 6120616 6793017 6243062 6816463  
 6366418 6939819 6489864 6052265 6502210  
 6175611 6625666 6298067 6748012 6311413  
 6861468 6434869 6984814 6557215 6007260  
 7781772 7231727 7804128 7354173 7927574  
 7477529 7040920 7590975 7163376 7613321  
 7286722 7736777 7309178 7859123 7422524  
 7972579 7545970 7095925 7668326 7118371  
 8892883 8342838 8915239 8465284 8038685  
 8588630 8151031 8601086 8274487 8724432  
 8397833 8847888 8410289 8960234 8533635  
 8083680 8656081 8106036 8779437 8229482  
 9903994 9453949 9026340 9576395 9149796  
 9699741 9262142 9712197 9385598 9835543  
 9408944 9958999 9521390 9071345 9644746  
 9194791 9767192 9217147 9880548 9330593

수열  $S_k = (s_{k,0}, s_{k,1}, \dots, s_{k,n-1})$ ,  $s_{k,0} = k$ ,  $k=0,1,\dots,T-1$ 를 생성한다. 여기서  $s_{k,i}$ 는  $k$ 번째 수열의  $i$ 번째 수를 표현한다. 정의에 의하여

$$s_{k,i+1} = (s_{k,i} + d_i) \bmod T, \quad s_{k,0} = k$$

이므로

$$s_{k,i+1} = (s_{k,0} + \sum_{t=0}^i d_t) \bmod T$$

이 성립한다. 그리고 임의의 정수  $m$  및  $l$ 에 대하여,  $s_{k,i} = (s_{k,0} + m) \bmod T$  이면  $s_{l,i} = (s_{l,0} + m) \bmod T$  가 성립한다. 그런데,  $s_{k,0} = k$  이고  $k$  에는 0에서  $T-1$ 까지의 모든 수가 차례로 대입되므로  $s_{k,i}$  에도 0에서  $T-1$ 까지의 모든 수가 대입된다. 그러므로 1차 차분수열에서 연속된 두 수의 조합  $(d_j, d_{j+1})$  이 존재한다면 다음과 같이 세 수의 조합  $(s_1, s_2, s_3)$  이 존재해야 한다.

$$\begin{aligned} s_1 & \\ s_2 &= s_1 + d_j \\ s_3 &= s_2 + d_{j+1} = s_1 + d_j + d_{j+1} \end{aligned}$$

그런데, 가정에서 세 수의 조합  $(s_1, s_2, s_3)$  이 존재하지 않는다고 하였으므로  $(d_j, d_{j+1})$  은 존재할 수 없다.

그렇다면, 마찬가지로

$$a_p = \begin{cases} d_{j+1} - d_j & \text{if } d_{j+1} \geq d_j \\ T + d_{j+1} - d_j & \text{otherwise} \end{cases}$$

를 만족하는  $a_p$  가 2차 차분 수열에 존재할 수 없다. 그 이유는 알고리즘의 단계2에서 정의 2에 따라  $T/(n-2)$  개 각각의 2차 차분수열  $A_k$  에 대하여  $T$  개의 서로 다른 1차 차분수열  $D_k = (d_{k,0}, d_{k,1}, \dots, d_{k,n-2})$ ,  $k=0,1,\dots,T-1$  을 생성하는데,  $d_{k,0} = k$  가 되도록 하므로 앞서와 마찬가지로  $d_{k,i}$  에도 0에서  $T-1$  까지의 모든 수가 대입되기 때문이다.

그런데, 단계 1에서  $T/(n-2)$  개의 서로 다른 2차 차분수열  $A_k$  를 생성할 때 모든 숫자를 반드시 한번씩 사용하였으므로 이에 모순이 된다. 따라서 본 알고리즘에 따라 작성된 목록에는 주어진 어휘 내에 포함된 모든 단어의 좌우 문맥이 고려된 조합이 하나도 빠짐없이 모두 존재한다.

#### 4.2. 결과의 효율성

본 알고리즘의 단계1에서는 어휘 세트에 포함된 단어가  $T$  개이고  $n$  연 숫자를 사용할 때  $T/(n-2)$  개의 2차 차분

### IV. 알고리즘의 분석

#### 4.1. 알고리즘의 완전성 (Completeness)

본 절에서는 알고리즘의 완전성을 증명하고자 한다. 여기서 증명하고자 하는 것은 본 알고리즘에 의해 작성된 목록에는 주어진 어휘 내에 포함된 모든 단어의 좌우 문맥이 고려된 조합이 하나도 빠짐없이 모두 존재한다는 것이다. 증명을 위해서 어떤 연속된 세 수의 조합  $(s_1, s_2, s_3)$  이 결과에서 누락되어 있다고 가정하고, 이것이 모순에 이를음을 보여준다.

만일 조합  $(s_1, s_2, s_3)$  이 존재하지 않는다면 1차 차분수열에서 다음을 만족하는 연속된 두 수의 조합  $(d_j, d_{j+1})$  이 존재하지 않아야 한다.

$$d_j = \begin{cases} s_{j+1} - s_j & \text{if } s_{j+1} \geq s_j \\ T + s_{j+1} - s_j & \text{otherwise} \end{cases}$$

그 이유는 다음과 같다. 알고리즘의 단계3에서 정의 1에 의하여  $T$  개 각각의 1차 차분수열에 대하여  $T$  개의 최종

수열을 생성한다. 그리고, 단계2에서는 각각의 2차 차분 수열을 이용하여 0에서  $T-1$ 까지의 수를 초기값으로 하는 1차 차분수열을 생성하므로 총 1차 차분수열의 수는  $T^2/(n-2)$ 개가 된다. 마지막으로 단계 3에서 각각의 1차 차분수열을 이용하여 0에서  $T-1$ 까지의 수를 초기값으로 하는 최종 수열을 생성하므로 총 결과 수열의 수는  $T^3/(n-2)$ 개가 된다. 또한, 총 좌우문맥종속 숫자는  $T^3$ 개가 되고,  $n$ 원 숫자를 사용하게 되면 한 문장에  $n-2$ 개의 좌우문맥종속 숫자음이 포함되므로 모든 좌우문맥종속 숫자음을 표현하려면 최소한  $T^3/(n-2)$ 개의 문장이 있어야 한다. 이것은 제안한 알고리즘이 생성하는 문장의 수와 일치하며, 4.1절에서 보인 바와 같이 누락된 문맥이 존재하지 않으므로, 제안한 알고리즘은 중복된 문맥을 만들지 않고 최적의 수열을 생성함을 알 수 있다.

단,  $T/(n-2)$ 가 정수가 아니라면 길이가  $n-2$ 보다 작은 2차 차분수열이 한개 만들어지게 되고, 이것은  $n-1$ 보다 작은 길이의 1차 차분수열  $T$ 개를 만들며, 다시 길이가  $n$ 보다 작은 최종 수열  $T^2$ 개를 만들게 된다. 이때, 길이가 작은 최종 수열을 다시 적절히 조합하여 재구성하면 보다 적은 개수의 길이가  $n$ 인 수열을 작성할 수 있게 된다.

### 4.3. 목록의 확장

2장에서는 하나의 단어에 대하여 좌우 1개씩의 문맥만을 고려한 발성목록을 작성하는 알고리즘을 제안하였지만, 이 알고리즘을 확장하면 좌우 문맥의 개수를 임의의 크기로 확장할 수 있다. 길이  $n$ 의 최종 숫자열( $s^n$ )을 차분 수열로 하여 다시 길이  $n+1$ 의 수열( $s^{n+1}$ )을 다음과 같이 만들 수 있다.

$$s_i^{n+1} = (s_0^{n+1} + \sum_{i=0}^{i-1} s_i^n) \bmod T$$

이와 같은 방법으로 반복하면 문맥의 길이를 자유롭게 확장할 수 있다.

## V. 결론

본 논문에서는 아직도 어려운 문제로 남아있는 연결 숫자음 인식기의 학습용 음성DB구축을 위해 효율적인 발성목록 작성 알고리즘을 제안하였다. 제안한 알고리즘은 모든 숫자음의 좌우 문맥을 고려한 환경이 모두 균일하게 포함되어 있는 최소한의 목록을 작성한다. 알고리즘에

의하여 생성된 목록에는 누락된 좌우문맥종속 숫자음이 없으며, 또한 중복된 좌우문맥종속 숫자음이 없으므로, 한 사람이 모든 환경의 숫자음을 발성하도록 할 수 있다. 또한, 난수 발생과 반복적인 최적화로 목록을 생성하는데 오랜 시간이 걸리는 기존의 방법과는 달리 빠른 시간에 원하는 목록을 얻을 수 있다. 또한, 2차 차분수열 내의 숫자의 순서를 변경함으로써 서로 다른 목록을 다수 만들어 낼 수 있다. 제안된 알고리즘을 사용하여 작성한 목록으로 수집한 음성DB를 사용하여 문맥종속 숫자음 모델을 학습하면 높은 연결 숫자음 인식 성능을 얻을 것을 기대할 수 있다.

## 감사의 글

이 논문은 2002년도 서울시립대학교 학술연구 조성에 의하여 연구되었음.

## 참고 문헌

1. Kwon, O. W. and Un, C. K., "Context-dependent word duration modeling for Korean connected digit recognition," *Electronics Letters*, 31 (19), 1630 -1631, 14 Sep. 1995.
2. Zhu Xuan, Li Husheng, Lin Jia and Liu Runsheng, "Efficient decoding algorithms for Mandarin connected digit speech recognition," *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*, 555-558, 2001.
3. <http://www ldc.upenn.edu/Catalog/docs/LDC93S10/tidigits.txt>
4. <http://www ldc.upenn.edu/Catalog/CatalogList/LDC93S64-4/PROMPTXT.TBL>
5. Cossi, P., Hosom, J.-P., Shalkwyk, J., Sutton, S., and Cole, R. A., "Connected digit recognition experiments with the OGI Toolkit's neural network and HMM-based recognizers," *Proceedings of IEEE 4th Workshop on Interactive Voice Technology for Telecommunications Applications, IVTTAETWR-98*, 135-140, Turin, Italy, September, 1998.

## 저자 약력

### • 유 하 진 (Ha-Jin Yu)



1990년, 1992, 1997년: KAIST 전산학과 (B.S., M.S., Ph.D)

1997년 1월 ~ 2000년 6월: LG전자기술원 선임연구원

2000년 7월 ~ 2002년 2월: SL2(주) 연구소장

2002년 3월 ~ 현재: 서울시립대학교 컴퓨터과학부 조교수

\* 주관심분야: 음성인식, 화자인식, 기계학습