# 뉴로-퍼지 소프트웨어 신뢰성 예측에 대한 최적의 데이터 분할비율에 관한 연구

이 상 운†


## 요 약

본 논문은 미래의 소프트웨어 고장 수나 고장시간 예측 정확성을 얻기 위해, 뉴로-퍼지 시스템을 이용할 경우 최적의 검증 데이터 할당 비율에 대한 연구이다. 훈련 데이터가 주어졌을 때, 과소 적합과 과잉 적합을 회피하면서 최적의 일반화 능력을 얻기 위해 Early Stopping 방법이 일반적으로 사용되고 있다. 그러나 훈련과 검증 데이터로 얼마나 많은 데이터를 할당할 것인가는 시행착오법을 이용해 경험적으로 해를 구해야만 하며, 과다한 시간이 소요된다. 최적의 검증 데이터 양을 구하기 위해 규칙 수를 증가시키면서 다양한 검증 데이터 양을 할당하였다. 실험 결과 최소의 검증 데이터로도 좋은 예측 능력을 보였다. 이 결과는 뉴로-퍼지 시스템을 소프트웨어 신뢰성 분야에 적용시 실질적인 지침을 제공할 수 있을 것이다.


# A Study of Optimal Ratio of Data Partition for Neuro-Fuzzy-Based Software Reliability Prediction

Sang-Un Lee†


## ABSTRACT

This paper presents the optimal fraction of validation set to obtain a prediction accuracy of software failure count or failure time in the future by a neuro-fuzzy system. Given a fixed amount of training data, the most popular effective approach to avoiding underfitting and overfitting is early stopping, and hence getting optimal generalization. But there is unresolved practical issues : How many data do you assign to the training and validation set? Rules of thumb abound, the solution is acquired by trial-and-error and we spend long time in this method. For the sake of optimal fraction of validation set, the variant specific fraction for the validation set be provided. It shows that minimal fraction of the validation data set is sufficient to achieve good next-step prediction. This result can be considered as a practical guideline in a prediction of software reliability by neuro-fuzzy system.


**키워드 : 뉴로-퍼지(Neuro-fuzzy), 일반화 능력(Generalization ability), 초기 종료(Early stopping), 검증 데이타(Validation data set), 평균상대오차(Average relative error)**

## 1. 서 론

시험과 운영 단계에서 획득된 고장 데이터로부터 미래의 소프트웨어 신뢰성을 예측하는데 일반적으로 소프트웨어 신뢰성 성장 모델 (SRGMs : Software Reliability Growth Models)이 사용되고 있다. 소프트웨어 신뢰성 모델의 예측력 (Predictability 또는 Generalization Ability)은 과거와 현재의 고장 발생 현상으로부터 미래의 고장 발생 현상을 예측할 수 있는 모델의 능력을 의미한다.

대부분의 SRGMs은 소프트웨어에 내재된 결함 특성과 고장의 통계적 발생 과정에 관한 가정에 기반을 두고 소프트웨어에 대한 전반적인 신뢰성 평가 및 예측을 제공한다. 또한 최적의 소프트웨어 양도 시점을 결정하는데도 사용된다. 이 부류의 모델들은 특정 소프트웨어의 개발 및 운영 환경에 대한 다양한 가정을 반영한 다수의 모수들 (Parameters)을 포함하고 있다. 그러나, 모든 소프트웨어 시스템에 적용 가능한 하나의 보편적인 모델이 없는 실정이다. 만약, 소프트웨어 시스템의 과거 고장 이력에 기반을 두고 자체의 모델을 개발하는 시스템을 갖고 있다면 그와 같은 가정은 고려하지 않아도 된다. 이 요구조건을 만족하는 것으로 신경망 또는 뉴로-퍼지 시스템이 있다. 이 시스템들의 특징은 사전 정보인 소프트웨어의 환경과 조건에 대한 가정을 고려하지 않는다. 단지, 관찰된 고장 이력만을 입력으로 요구하며, 관찰된 표본 데이터에 적합한 함수를 학습할 수 있는 능력을 갖고 있으며, 보편적 함수 근사기 (Universal Function Approximator)로서의 기능을 수행할 수 있음이 증명되었다.

† 정 회 원 : 국방품질관리소 항공전자장비 및 소프트웨어 품질보증 담당
논문접수 : 2001년 2월 26일, 심사완료 : 2001년 4월 24일

뉴로-퍼지 시스템은 관찰된 표본 데이터를 처리하여 시스템의 맴버쉽 함수의 형태 모수를 결정하기 위해 신경망 이론에서 유도된 학습 알고리즘을 사용하는 퍼지 시스템으로 함수근사와 패턴분류에 널리 적용되고 있다. 뉴로-퍼지 시스템에 대한 자세한 설명은 Lauck [1]과 Nauck et al. [2]를 참조하기 바란다. 또한 뉴로-퍼지 시스템을 소프트웨어 신뢰성 예측분야에 적용한 사례는 이상운 [3]이 있다. 주어진 문제에 대해 적절한 뉴로-퍼지 시스템을 확인하기 위해서는 맴버쉽 함수와 최적의 규칙수가 정의되어야만 한다. 이는 사전 지식에 의해, 학습에 의해, 또는 이 2가지를 합성한 방법에 의해 수행된다. 소프트웨어 신뢰성을 예측하기 위해 뉴로-퍼지 시스템을 적용시, 규칙의 개수와 데이터 분할 비율에 대한 적절한 정보를 갖고 있다면, 다양한 모델을 적용하여 수행시킨 결과를 비교해 최적의 모델을 선정하는 시행착오법에 소요되는 노력을 감소시킬 수 있다. 따라서, 본 논문에서는 최적의 모델 구조에 대한 주어진 데이터의 적절한 분할 비율이 뉴로-퍼지 시스템을 적용한 소프트웨어 신뢰성 예측력에 어떠한 영향을 미치는지 실험을 통해 지침을 제시하고자 한다. 2장에서는 모델의 구조와 데이터 크기가 일반화 오차에 미치는 영향과 관련 연구에 대해 언급한다. 3장에서는 데이터 분할 비율이 소프트웨어 신뢰성 예측력에 미치는 영향에 대해 실험을 통해 비교 분석하여 본다.

## 2. 모델의 예측력(일반화 능력)

주어진 문제에 적합한 특정 모델을 선택하는데 있어서 가장 중요한 점은 선정된 모델이 얼마나 일반화 능력(Generalization Ability)을 갖고 있는가 이다. 이는 모델을 훈련시키는데 사용된 훈련 데이터에 존재하지 않는 새로운 데이터에 대해 모델이 얼마나 잘 적합할 수 있는지의 능력이다. 즉, 우리의 목적은 새로운 데이터에 대해 최상의 성능을 가진 모델을 찾는 것이다[4]. 따라서, 본 장에서는 주어진 문제에 대해 최적의 일반화 능력을 갖는 모델을 선정하기 위해, 일반화 능력에 영향을 미치는 요인이 어떤 것이 있는지, 어떤 방법을 사용하는 것이 최선인지를 고찰해 보고자 한다.

### 2.1 예측력에 영향을 미치는 요인

모집단 (주어진 문제)에서 추출한 표본 (훈련 데이터)을 사용해 우리가 얻고자 하는 최적의 실제 함수에 근사 시키는 경우를 고려해 보자. 모델의 함수 근사 오차 (Approximation Error) $E_a^*$는 근사적으로 식 (1)과 같이 표현된다.

$$E_a^* \approx E_r + E_g \quad (1)$$

여기서 $E_r$은 표현오차 (Representation Error)이며, $E_g$는 일반화 오차 (Generalization Error)이다. 표현오차 ($E_r$)는 함수에 대해 완전한 지식 (잡음이 없는 무한한 학습 데이터)을 갖고 있을 경우 주어진 함수를 표현할 수 있도록 얼마나 정확한 구조를 제공할 수 있는가를 알려준다. 구조가 충분히 복잡하지 않으면 복잡한 데이터에 내재된 신호를 완전히 검출하지 못해 과소 적합 (Underfitting)을 유발시킨다. 이에 반해 구조가 너무 복잡해지면 잡음에 까지도 적합되어 새로운 데이터에 대한 일반화 능력을 무시하는 과잉 적합 (Overfitting)을 유발시킨다. 과소 적합은 출력에 과다한 편향 (Bias)을, 과잉 적합은 과다한 편차 (Variance)를 나타내며 이를 편향-편차 문제라 한다[5]. 일반화 오차 ($E_g$)는 실제로는 항상 유한한 학습 데이터를 갖고 있으므로 제한된 학습 데이터 크기로 최상의 근사 함수를 추정함으로 인해 발생되는 추가적인 오차로 학습 데이터의 크기 ($N$)가 증가하면 0로 수렴된다.

따라서, 모델의 근사 오차는 모델의 구조와 주어진 데이터의 크기에 의존함을 알 수 있다. 만약 모델의 구조가 주어진다면 (규칙 수 결정), 표현오차 ($E_r$)는 $N$개의 학습 데이터의 선택과 모델의 맴버쉽 함수에 대한 초기 값에 의존함을 알 수 있다.

일반적으로, 주어진 입출력에 대해 원하는 정확도를 가진 관계를 찾는데 충분한 정보를 제공하기 위해 얼마나 많은 학습 데이터가 필요한가와 얼마나 많은 규칙수가 사용되어야 하는가의 문제는 근사 시킬 함수의 복잡도에 의존하며, 일반적인 해답은 결코 존재하지 않는다. 소프트웨어 시험을 수행하는 과정에서 수집된 고장 데이터는 유한개이며, 고정되어 있다. 또한 이 데이터를 사용해 최적의 모델을 선정해 미래의 예측력을 평가해야만 한다. 따라서, 이 경우는 학습 데이터의 크기가 고정된 경우로 최적의 모델의 구조 (규칙수)만 결정하면 된다.

### 2.2 모델 구조

뉴로-퍼지 시스템에서 모델의 구조, 즉 최적의 규칙 수를 결정하는 방법을 살펴보자. 주어진 데이터로부터 Fuzzy C-Means 알고리즘 이용 출력 데이터를 분석하여 필요한 초기 규칙 수를 결정하고 입력 변수 식별과 입력 맴버쉽 함수를 결정하기 위해 신경망의 학습 알고리즘을 사용하는 방법이 있다[1]. 또한 사전 정보를 갖고 있지 못할 경우, 규칙이 없는 상태에서 사전 정보를 가진 경우의 획득된 규칙수로 초기화시키고 구성 알고리즘을 적용하여 나머지 규칙을 추가시켜 최적의 규칙 수를 찾는 방법도 제시되어 있다[2]. 그러나 이들 방법 중 최적의 구조를 찾는데 가장 효과적인 방법이라고 제시된 것은 없다. 단지 주어진 문제의 복잡한 발생 과정에 영향을 받는다.

### 2.3 데이터 분할

모델을 훈련시키기 위해 수집된 데이터를 단순히 훈련

데이터와 시험 데이터로 분할하는 Split-Sample Validation 방법이 있다. 신경망이나 뉴로-퍼지 분야에서는 일반적으로 관찰된 데이터를 훈련 데이터, 검증 데이터와 시험 데이터로 분류한다. 훈련 데이터는 모델의 모수 (맴버쉽 함수)를 적합(Fit) 시키기 위해 학습에 사용되는 표본 데이터이다. 검증 데이터는 모델의 모수를 조율 (규칙 수 결정)하는데 사용되는 표본으로 훈련에는 참여하지 않고 단지 언제 시험을 종료시킬 것인가를 결정하는데 사용된다. 또한 시험 데이터는 훈련과 검증 데이터를 이용해 확정된 모델에 대한 일반화 성능만을 평가하는데 사용되는 표본 데이터로 일반화하고자 하는 모집단을 대표하는 표본이어야 한다. 따라서, 훈련 데이터와 검증 데이터를 이용해 훈련이 종료된 모델에 시험 데이터로 수행시키면 시험 데이터의 오차는 일반화 오차의 불편 추정치를 제공한다[6-8]. 이 방법의 단점은 훈련과 검증에 사용 가능한 데이터의 양이 줄어든다는 점이다. 훈련 데이터를 사용해 모델을 학습시킬 경우 언제 훈련을 종료시킬 것인가가 문제가 된다. 전형적으로 모델의 학습과정에서 훈련 데이터에 대한 오차는 계속적으로 감소하는데 반해 검증 데이터의 오차는 점차 감소하다가 특정 시점에서 급격히 증가하는 경향을 나타낸다. 이러한 현상을 감안하여 검증 데이터의 오차가 급격히 증가하는 초기 시점에서 모델의 훈련을 멈추는 기법이 있다. 이 시점에서 시험 데이터 집합의 일반화 오차가 최소화되어 모델의 일반화 능력이 최대로 됨이 밝혀졌으며, 이 전략이 Early Stopping (Stopped Early, Optimal Stopping)기법이다.

신경망 분야에서 주어진 학습 데이터 크기와 은닉 뉴런 수와의 관계에 관한 연구는 Vyśniauskas [5], Baum et al. [7], Barron [9, 10], White [11]가 있다. 충분한 데이터 크기가 주어진다면 추정오차를 감소시킬 수 있어 일반화 오차도 감소가 된다. 그러나 소프트웨어 시험을 수행하는 과정에서 미래의 신뢰성을 예측하는 경우를 살펴보자. $i$번째 시험시간 $t_i$까지 발견된 누적 고장 수 $m_i$ 또는 $i$번째 고장이 발생한 시간 $s_i$로 수집된 고장 데이터를 이용할 경우, 학습 데이터의 크기는 유한한 값을 가진다. 이 경우 충분한 데이터 크기를 제공하질 못해 일반화 오차를 어느 한계 이하로는 감소시킬 수 없다. 단지 Early Stopping 기법을 적용시 관찰된 데이터를 어떤 비율로 훈련 데이터와 검증 데이터로 분할시킬 것인가에 따라 예측 오차에 영향을 미친다. 훈련과 검증 데이터의 분할 비율에 대한 연구는 Amari et al. [6], Bös [8]과 Barber et al. [12]이 있다. 수집된 데이터에서 검증 데이터에 사용될 최적의 표본 비율 $\eta^{opt}$를 구하기 위해 Amari et al. [6], Barber et al. [12]은 주어진 데이터가 선형함수인 경우 식 (2)와 같이 입력변수의 개수에 비례하는 식을 유도하였다.

$$\eta^{opt} \propto N^{\tau} \tag{2}$$

여기서, $N$은 입력 변수의 개수이며, $\tau$은 상수로 Amari et al. [6]는 1/2로, Barber et al. [12]는 1/3로 설정하였다. Bös [8]은 비선형 함수인 tanh 함수를 학습하는 경우에 대해, 훈련 데이터가 400개, $\eta$는 0.20, 0.10과 0.05에 대해 실험 결과 $\eta^{opt}$를 찾는 것은 매우 어려우며 단지 적은 비율의 검증 데이터로도 좋은 결과를 얻는데 충분함을 제안하였다. Haykin [13]는 통계학 분야에서는 전형적으로 학습 데이터의 약 10~20%를 검증 데이터로 사용함을 제시하였다. 그러나 위의 연구사례 모두 신경망에 대한 연구이며, 뉴로-퍼지 시스템에 관한 연구는 없는 실정이다. 또한 데이터의 양이 보통 또는 충분히 큰 경우이고, 선형이나 tanh인 특정 함수에 대한 실험 결과임을 알 수 있다.

소프트웨어는 다양하며, 고장 발생 과정 또한 비선형으로 모두 다른 함수 형태를 취한다. 따라서 특정 함수에 대한 연구 결과를 적용할 경우 오차가 발생할 수 있다. 또한 소프트웨어 시험시 수집된 Lyu [14]의 데이터 33개를 분석한 결과는 <표 1>에 제시된 바와 같이 매우 적은 수이다.

〈표 1〉 고장 수와 고장시간 데이터.

| 소프트웨어 | 구 분 | 데이터 크기 | 소프트웨어 | 구 분 | 데이터 크기 |
|---|---|---|---|---|---|
| CSR1 | 고장시간 | 397 | Dataset2 | 고장시간 | 111 |
| CSR2 | 고장시간 | 129 | Dataset4 | 고 장 수 | 32 |
| CSR3 | 고장시간 | 104 | Dataset5 | 고 장 수 | 14 |
| Data1 | 고장시간 | 14 | J1 | 고 장 수 | 62 |
| Data3 | 고 장 수 | 46 | J2 | 고 장 수 | 181 |
| Data4 | 고 장 수 | 17 | J3 | 고 장 수 | 41 |
| Data5 | 고 장 수 | 10 | J4 | 고 장 수 | 114 |
| Data6 | 고 장 수 | 13 | J5 | 고 수 | 73 |
| Data7 | 고 장 수 | 109 | OCD1 | 고 장 수 | 102 |
| Data8 | 고 장 수 | 111 | S2 | 고장시간 | 54 |
| Data9 | 고 장 수 | 199 | S27 | 고장시간 | 41 |
| Data10 | 고 장 수 | 16 | SS1 | 고 장 수 | 81 |
| Data11 | 고장시간 | 118 | SS4 | 고장시간 | 197 |
| Data12 | 고장시간 | 180 | SYS1 | 고장시간 | 135 |
| Data13 | 고장시간 | 213 | SYS2 | 고장시간 | 86 |
| Data14 | 고 장 수 | 46 | SYS3 | 고장시간 | 207 |
| Dataset1 | 고 장 수 | 140 | | | |

따라서, 복잡한 비선형 형태를 취하는 소프트웨어 신뢰성 분야에서, 훈련 데이터와 검증 데이터의 분할 비율이 모델의 예측력에 미치는 영향에 대한 정형화된 관계를 제시할 필요가 있다. 정형화된 결과를 사용할 경우 여러 모델을 실험하여 적합한 모델을 결정하는데 소요되는 노력을 감소시킬 수 있을 것이다.

## 3. 예측 실험 및 결과

본 장에서는 주어진 고장 데이터로부터 뉴로-퍼지 소프트웨어 신뢰성 모델의 예측력에 영향을 미치는 모델의 구조와 학습 데이터 분할 비율과의 관계를 실험을 통해 검증한다. 즉, 모델의 구조를 변경시켜 가면서, 주어진 데이터를 훈련과 검증 데이터의 다양한 비율로 설정하여 훈련시킨 결과 최적의 구조를 결정한다. 또한, 이 구조에 대한 데이

터의 분할 비율에 따른 모델의 예측력을 분석하고자 한다.

소프트웨어 시험 중 $i$시점까지 수집된 고장 데이터를 이용해 이후의 $r$시점에서의 누적 고장 수 ($m_i$) 또는 고장 발생시간 ($s_i$)을 예측하는 경우를 고려해 보자. Karunanithi et al. [15, 16]은 예측력 척도로 $i$에 대해 다음 단계 (Next-Step)와 마지막 단계 (End-Step) 예측 상대 오차를 고려하였다. 미래의 몇 단계 앞을 예측할 것인가는 주관적 관점에 따라 차이가 발생되나 마지막 단계 예측은 데이터의 개수에 따라 차이가 발생한다. 또한, 실제 적용에 있어서도 시험을 종료하는 시점을 명확히 판단할 수 없다. 따라서, 우리의 주요 관심은 소프트웨어가 주어진 시간 동안 (다음 단계) 고장 없이 작동하는가 여부이므로 본 연구에서는 다음 단계 예측에 한정하여 모델의 예측력을 분석하여 본다. 그러나 다양한 $r$ 이 적용 가능하다.

적절한 데이터 분할 비율을 결정하기 위해, 식 (1)에서 $\eta$의 범위를 $0<\eta\leq 0.5$로 설정하여 $\eta=0.05,\ 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5$에 대해 각각 실험을 수행한다. 모델의 예측력은 훈련 데이터 수, 잡음의 크기, 학습하고자 하는 함수의 복잡도와 모델의 구조에 큰 영향을 받기 때문에 주어진 데이터로부터 직접 모델의 구조를 결정할 수 있는 방법은 없다. 따라서, 규칙의 개수를 직접 결정하지 않고 2개~40개까지 변화시켜 가면서 모델에 대한 최적의 구조를 결정한다. 시스템의 출력과 목표 값 (Target)간의 오차가 최소화되도록 LMS (Least Mean Square error) 알고리즘을 사용하여 시스템을 학습시킨다. 시스템의 훈련을 종료하는 시점을 결정하기 위해 Early Stopping 기법을 사용한다. 시스템의 훈련제도로는 Karunanithi et al. [15, 16]가 제안한 <표 2>의 훈련제도를 사용한다.

〈표 2〉 훈련제도

| 훈 련 제 도 | 고장수 데이터 | | 고장시간 데이터 | |
|---|---|---|---|---|
| | 입력 | 출력 | 입력 | 출력 |
| 일 반 훈 련 제 도 (Generalization Regime) | $t_i$ | $m_i$ | $i$ | $s_i$ |
| 예 측 훈 련 제 도 (Prediction Regime) | $t_{i-1}$ | $m_i$ | $i-1$ | $s_i$ |

즉, 고장 수 데이터의 일반훈련제도는 수집된 데이터에 대해 $i$시점의 시험시간 $t_i$을 입력으로 하였을 때 $i$시점까지 발견된 누적 고장 수 $m_i$가 출력되도록 훈련을 시키는 것이고, 훈련이 종료된 후 $i+1$시점의 시험시간 $t_{i+1}$을 입력하면 이 시점에서의 예측되는 누적 고장 수 $m_{i+1}$을 얻을 수 있다. 선택된 데이터에 대해 2개 훈련제도 각각에 대해 규칙 수와 데이터 분할 비율에 대해 뉴로-퍼지 소프트웨어 신뢰성 예측 모델을 구축한다. 다음 단계 예측에는 <표 1>에서 기술된 19개의 고장 수 데이터와 14개의 고장시간 데이터에서 실험에 사용될 데이터를 <표 3>과 같이 선택하였다.

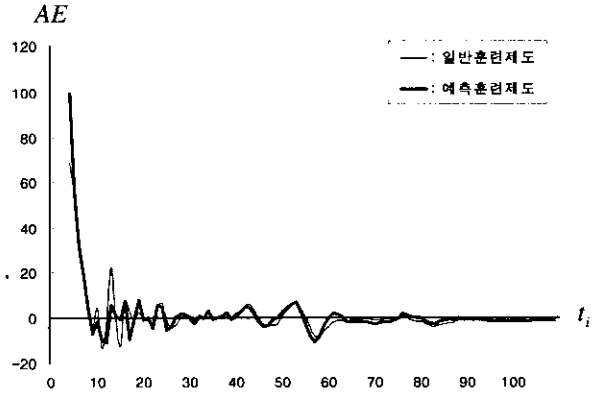
〈표 3〉 실험에 사용된 데이터

| 고장 데이터 크기 | 소 프 트 웨 어 | |
|---|---|---|
| | 고장 수 데이터 | 고장시간 데이터 |
| 10 ~ 50 | Data14, J3 | Data1, S27 |
| 51 ~ 100 | J5, SS1 | S2, SYS2 |
| 101 ~ 150 | Data7, Dataset1 | CSR2, SYS1 |
| 151 ~ 200 | Data9, J2 | Data12, SS4 |
| 201 ~ 250 | - | Data13, SYS3 |

$n$을 고장 데이터의 총 개수라 할 때, 주어진 고장 수 데이터 $(t_i, m_i), i=1,2,\cdots,n$에 대해, $l$을 뉴로-퍼지 시스템 훈련에 사용된 데이터 크기로, $r$은 예측 대상 시점으로 가정하자. 훈련에 사용되는 최소한의 데이터 크기는 3으로 설정한 후(즉, $l=3$) 주어진 훈련제도에 대해 다음 단계 예측 절차는 다음과 같이 수행된다.

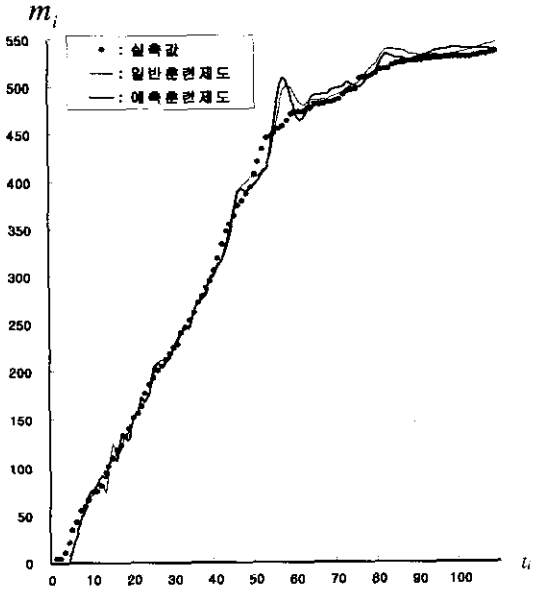
단계 (1) 훈련 제도와 규칙 수 결정.
단계 (2) 검증 데이터 비율 $\eta$ 결정.
단계 (3) 학습 알고리즘과, 훈련 데이터 $(t_i, m_i)$, $i=1,2,\cdots,l$을 사용하여 훈련 수행.
단계 (4) $t_{l+j}, j=1,2,\cdots,r$ 시간을 입력하여 $j$ 번째 미래의 누적 고장 수, $\hat{m}_{l+j}$, 예측과 상대 오차 $e_{lj}=\frac{(\hat{m}_{l+j}-m_{l+j})}{m_{l+j}}\cdot 100$ 계산.
단계 (5) $l$을 1 증가, $l=n-r$이 될 때까지 (2)~(4) 단계 수행.
단계 (6) $l$ 에 대해 $e_{lj}$ 의 평균 계산.

$l$에 대한 $e_{lj}$의 평균은 $\bar{e}_{\cdot j}$로 표기되며, 이는 평균 상대 예측 오차 (the average relative prediction error : AE) 이다. 훈련제도 각각에 대해 규칙 수와 검증 데이터 비율 $\eta$를 증가시키면서 단계 (2)~(6)이 수행하였다. 다음 단계 예측 값인 출력과 AE를 구하고, 시스템의 다음 단계 평균 예측 오차가 최소가 되는 규칙 수를 선택하였다.




(그림 1) Data7에 대한 AE ( $\eta$ = 0.05)

Data7에 대해 최적인 규칙을 사용한 경우, 최적의 $\eta$에 대한 AE를 (그림 1)에, 출력은 (그림 2)에 제시하였다. <표 4>는 8개의 고장 데이터에 대해 선택된 최적의 규칙 수와 AE를 제시하고 있다.

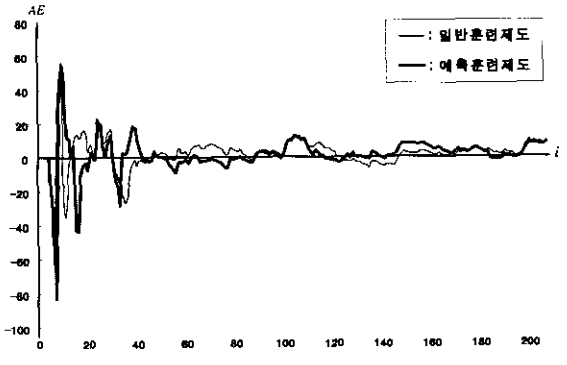



(그림 2) Data7 누적 고장수 ( $\eta$ = 0.05)

〈표 4〉 고장 수 데이터의 최적의 규칙 수와 AE

| System/ 훈련제도 | | 구분 | $\eta$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0.5 | 0.45 | 0.4 | 0.35 | 0.3 | 0.25 | 0.2 | 0.15 | 0.1 | 0.05 |
| Data 14 | 일반 | 규칙수 | 30 | 30 | 30 | 29 | 27 | 28 | 27 | 30 | 30 | 6 |
| | | AE | 13.13 | 13.27 | 13.22 | 10.89 | 10.66 | 10.02 | 9.53 | 13.22 | 8.47 | 7.52 |
| | 예측 | 규칙수 | 30 | 30 | 30 | 30 | 30 | 29 | 27 | 26 | 26 | 29 |
| | | AE | 13.70 | 13.86 | 13.26 | 12.99 | 12.18 | 11.82 | 10.36 | 9.83 | 8.71 | 7.47 |
| J3 | 일반 | 규칙수 | 30 | 30 | 8 | 8 | 9 | 9 | 11 | 12 | 3 | 14 |
| | | AE | 9.74 | 11.59 | 10.43 | 10.99 | 11.06 | 9.74 | 9.32 | 7.50 | 6.25 | 4.93 |
| | 예측 | 규칙수 | 29 | 28 | 28 | 4 | 11 | 4 | 12 | 11 | 12 | 13 |
| | | AE | 11.76 | 13.58 | 15.06 | 13.66 | 12.03 | 10.80 | 9.04 | 7.11 | 5.99 | 6.01 |
| J5 | 일반 | 규칙수 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 |
| | | AE | 13.13 | 11.68 | 10.98 | 9.71 | 8.36 | 7.55 | 6.36 | 6.02 | 5.13 | 4.23 |
| | 예측 | 규칙수 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| | | AE | 10.37 | 10.01 | 8.87 | 7.61 | 8.61 | 7.60 | 6.95 | 6.43 | 5.33 | 4.61 |
| SS1 | 일반 | 규칙수 | 14 | 14 | 14 | 16 | 14 | 14 | 14 | 17 | 17 | 5 |
| | | AE | 11.09 | 9.87 | 9.82 | 9.20 | 7.37 | 7.28 | 6.39 | 5.72 | 4.53 | 3.40 |
| | 예측 | 규칙수 | 15 | 16 | 16 | 16 | 13 | 13 | 7 | 6 | 13 | 20 |
| | | AE | 21.54 | 18.43 | 16.24 | 12.65 | 10.78 | 8.58 | 7.33 | 6.32 | 5.21 | 2.99 |
| Data 7 | 일반 | 규칙수 | 2 | 6 | 6 | 3 | 3 | 3 | 6 | 6 | 5 | 5 |
| | | AE | 23.74 | 23.04 | 21.16 | 17.70 | 13.59 | 11.22 | 9.23 | 7.35 | 5.50 | 4.10 |
| | 예측 | 규칙수 | 12 | 6 | 8 | 3 | 3 | 3 | 12 | 12 | 6 | 6 |
| | | AE | 22.98 | 22.30 | 20.57 | 16.39 | 13.28 | 11.17 | 9.12 | 7.26 | 5.64 | 4.42 |
| Data set1 | 일반 | 규칙수 | 2 | 2 | 2 | 2 | 4 | 4 | 6 | 3 | 3 | 8 |
| | | AE | 15.93 | 15.52 | 13.22 | 11.94 | 12.44 | 10.07 | 9.27 | 6.85 | 5.58 | 4.06 |
| | 예측 | 규칙수 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 |
| | | AE | 16.29 | 14.93 | 13.10 | 10.53 | 10.11 | 8.62 | 6.47 | 5.62 | 4.26 | 3.45 |
| Data 9 | 일반 | 규칙수 | 16 | 17 | 17 | 17 | 6 | 6 | 6 | 6 | 6 | 5 |
| | | AE | 45.49 | 42.17 | 37.67 | 35.83 | 33.27 | 29.57 | 25.04 | 18.58 | 13.04 | 8.11 |
| | 예측 | 규칙수 | 16 | 17 | 17 | 27 | 6 | 6 | 6 | 6 | 6 | 5 |
| | | AE | 42.53 | 41.98 | 37.28 | 35.56 | 31.88 | 29.33 | 24.70 | 18.79 | 13.08 | 8.17 |
| J2 | 일반 | 규칙수 | 3 | 10 | 10 | 11 | 10 | 25 | 26 | 30 | 23 | 32 |
| | | AE | 36.99 | 36.67 | 36.27 | 35.37 | 31.02 | 27.72 | 22.26 | 17.55 | 9.45 | 6.35 |
| | 예측 | 규칙수 | 3 | 8 | 10 | 11 | 10 | 23 | 10 | 28 | 10 | 10 |
| | | AE | 35.82 | 36.50 | 35.14 | 33.37 | 30.11 | 26.20 | 21.72 | 17.40 | 12.84 | 8.42 |

주어진 고장시간 데이터 $(i, s_i), i=1,2,\cdots,n$에 대해 다음 단계 예측 절차는 고장 수 데이터와 동일한 과정으로 수행되었다. Sys3에 대해 최적인 규칙 수를 사용한 경우, 최적의 $\eta$에 대한 AE를 (그림 3)에 표시하였다. 최적의 규칙 수와 AE는 <표 5>에 표현되어 있다.




(그림 3) Sys3에 대한 AE ( $\eta$ = 0.05)

〈표 5〉 고장시간 데이터의 최적의 규칙 수와 AE

| System/ 훈련제도 | | 구 분 | $\eta$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0.5 | 0.45 | 0.4 | 0.35 | 0.3 | 0.25 | 0.2 | 0.15 | 0.1 | 0.05 |
| Data 1 | 일반 | 규칙수 | 13 | 23 | 23 | 9 | 9 | 9 | 15 | 19 | 15 | 15 |
| | | AE | 9.06 | 9.47 | 8.98 | 9.52 | 7.32 | 6.45 | 6.31 | 7.99 | 7.17 | 7.17 |
| | 예측 | 규칙수 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| | | AE | 10.27 | 10.27 | 9.37 | 10.80 | 11.89 | 10.36 | 9.38 | 9.45 | 8.46 | 7.62 |
| S27 | 일반 | 규칙수 | 5 | 5 | 5 | 5 | 5 | 5 | 2 | 12 | 16 | 5 |
| | | AE | 28.54 | 26.64 | 22.31 | 21.54 | 21.39 | 24.05 | 24.51 | 19.65 | 17.73 | 15.82 |
| | 예측 | 규칙수 | 5 | 20 | 21 | 23 | 2 | 3 | 2 | 2 | 15 | 5 |
| | | AE | 31.58 | 32.42 | 33.33 | 33.07 | 30.36 | 26.39 | 22.78 | 19.77 | 15.96 | 12.15 |
| S2 | 일반 | 규칙수 | 2 | 3 | 3 | 2 | 2 | 2 | 2 | 3 | 10 | 3 |
| | | AE | 36.40 | 30.80 | 26.81 | 22.74 | 19.41 | 17.08 | 15.78 | 26.81 | 13.45 | 9.93 |
| | 예측 | 규칙수 | 7 | 7 | 5 | 3 | 2 | 2 | 2 | 6 | 6 | 2 |
| | | AE | 30.46 | 27.70 | 23.58 | 22.10 | 17.95 | 15.12 | 15.24 | 12.69 | 10.48 | 8.27 |
| SYS 2 | 일반 | 규칙수 | 11 | 10 | 10 | 14 | 14 | 15 | 16 | 5 | 5 | 5 |
| | | AE | 15.84 | 15.98 | 14.68 | 13.26 | 12.69 | 11.23 | 10.80 | 8.59 | 7.64 | 6.70 |
| | 예측 | 규칙수 | 9 | 10 | 10 | 16 | 14 | 22 | 5 | 5 | 2 | 2 |
| | | AE | 12.63 | 12.15 | 12.75 | 12.06 | 10.68 | 10.78 | 9.66 | 8.04 | 7.33 | 6.62 |
| CSR 2 | 일반 | 규칙수 | 30 | 27 | 26 | 24 | 28 | 20 | 2 | 2 | 2 | 2 |
| | | AE | 34.84 | 34.37 | 33.02 | 30.60 | 28.66 | 24.62 | 21.78 | 17.67 | 14.78 | 11.89 |
| | 예측 | 규칙수 | 30 | 30 | 28 | 28 | 20 | 2 | 2 | 2 | 2 | 2 |
| | | AE | 35.26 | 35.29 | 33.61 | 31.46 | 29.21 | 26.40 | 21.88 | 18.03 | 15.23 | 11.87 |
| SYS 1 | 일반 | 규칙수 | 8 | 8 | 9 | 9 | 13 | 18 | 20 | 7 | 24 | 9 |
| | | AE | 27.50 | 24.92 | 23.47 | 19.60 | 18.49 | 16.18 | 15.13 | 11.38 | 9.41 | 7.44 |
| | 예측 | 규칙수 | 9 | 9 | 9 | 7 | 7 | 7 | 7 | 28 | 26 | 9 |
| | | AE | 25.40 | 21.60 | 18.95 | 17.75 | 16.94 | 15.16 | 13.47 | 7.65 | 9.19 | 3.29 |
| Data 12 | 일반 | 규칙수 | 2 | 28 | 30 | 28 | 29 | 2 | 2 | 2 | 2 | 2 |
| | | AE | 31.09 | 27.78 | 24.68 | 21.64 | 20.13 | 17.81 | 16.03 | 14.36 | 12.36 | 9.89 |
| | 예측 | 규칙수 | 29 | 29 | 29 | 29 | 30 | 29 | 2 | 2 | 23 | 2 |
| | | AE | 28.36 | 24.97 | 21.06 | 19.23 | 19.24 | 18.49 | 16.44 | 14.62 | 12.31 | 9.99 |
| SS4 | 일반 | 규칙수 | 23 | 22 | 23 | 2 | 2 | 5 | 5 | 2 | 7 | 2 |
| | | AE | 18.19 | 17.74 | 16.78 | 14.81 | 13.86 | 12.74 | 10.61 | 9.37 | 8.03 | 6.70 |
| | 예측 | 규칙수 | 23 | 21 | 22 | 2 | 2 | 5 | 5 | 5 | 7 | 2 |
| | | AE | 8.67 | 8.85 | 9.21 | 15.19 | 14.58 | 12.64 | 11.20 | 10.05 | 8.31 | 6.58 |
| Data 13 | 일반 | 규칙수 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| | | AE | 23.08 | 20.12 | 17.29 | 14.50 | 9.96 | 10.17 | 8.77 | 7.35 | 6.19 | 5.01 |
| | 예측 | 규칙수 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| | | AE | 24.12 | 20.90 | 17.93 | 15.32 | 13.11 | 10.60 | 9.12 | 7.68 | 6.51 | 5.01 |
| SYS 3 | 일반 | 규칙수 | 16 | 21 | 25 | 28 | 15 | 16 | 30 | 28 | 30 | 16 |
| | | AE | 15.12 | 13.81 | 12.82 | 11.84 | 10.42 | 9.96 | 9.23 | 8.17 | 7.20 | 6.23 |
| | 예측 | 규칙수 | 18 | 19 | 23 | 21 | 15 | 21 | 21 | 28 | 6 | 3 |
| | | AE | 12.99 | 12.91 | 12.29 | 11.60 | 10.37 | 9.61 | 9.03 | 8.47 | 6.50 | 4.53 |

실험 결과, Data1의 일반 훈련제도를 제외하고는 고장 수와 고장시간 데이터 모두 검증 데이터에 할당되는 비율이 최소가 될 때 (즉, $\eta=0.05$인 경우), 다음단계 예측력이 가장 좋은 결과를 얻었다. 따라서, 뉴로-퍼지 시스템을 활용하여 소프트웨어 신뢰성을 예측하는데 있어서 본 실험결과를 이용할 경우 시스템에 적용되는 데이터의 분할 비율의 적절성을 판단하는데 지침을 제공할 수 있을 것이다.

## 4. 결론 및 향후 연구과제

본 논문은 관찰된 고장 데이터로부터 소프트웨어 신뢰성을 예측하기 위해 뉴로-퍼지 시스템 활용시 훈련에 사용되는 데이터인 훈련 데이터와 검증 데이터의 최적의 분할비율에 관한 연구를 수행하였다. 본 연구에는 데이터 크기가 각각 다른 8개의 고장 수 데이터와 10개의 고장시간 데이터가 사용되었다. 실험 결과 다음단계 예측에 있어 검증 데이터에 대한 할당 비율을 최소화하는 것이 최적의 예측 결과를 나타냄을 알 수 있었다. 따라서, 뉴로-퍼지 시스템을 활용해 신뢰성을 예측할 때 예측력에 영향을 미치는 시스템 구조와 데이터 크기 문제가 있다. 데이터 크기는 주어진 데이터에서 최소 데이터를 검증 데이터로 할당하고 훈련 데이터 크기를 최대한으로 크게 하는 것이 다음 단계 소프트웨어 신뢰성 예측에는 적합함을 알 수 있었다. 본 실험결과는 뉴로-퍼지 시스템을 활용해 소프트웨어의 신뢰성을 예측하고자 할 때 데이터 분할 비율에 대한 지침으로 활용가능하다. 또한, 본 결과를 이용시 데이터 분할에 관한 시행착오법에 소요되는 시간과 노력을 줄일 수 있을 것이다.

예측력을 향상시키는데 있어서 또 하나의 문제점은 시스템 구조로 주어진 문제에 적절한 구조를 사전에 어떻게 인지할 수 있는가 이며, 이는 주어진 데이터를 분석함으로서 획득할 수 있을 것이다. 따라서 이 분야에 관해 추후 연구가 수행될 것이다.

## 참 고 문 헌


[1] D. Nauck, "Neuro-Fuzzy Systems : Review and Prospects," Proc. 5th European Congress on Intelligent Techniques and Soft Computing (EUFIT '97), Aachen, pp.1044-1053, 1997.

[2] D. Nauck and R. Kruse, "A Neuro-Fuzzy Approach to Obtain Interpretable Fuzzy Systems for Function Approximation," Proc. IEEE Conf. on Fuzzy Systems, Anchorage, AK, pp. 1106-1111, 1998.

[3] 이상운, "뉴로-퍼지 소프트웨어 신뢰성 예측," 정보과학회 논문지(B), 제27권 제4호, pp.393-401, 2000.

[4] C. M. Bishop, "Neural Networks for Pattern Recognition," Oxford University Press, 1995.

[5] V. Vyšniauskas, F. C. A. Groen, and B. J. A. Krose, "The Optimal Number of Learning Samples and Hidden Units in Function Approximation with a Feedforward Network," Technical Report, Dept. of Comp. Sys, Univ. of Amsterdam, CS-93-15, 1993.

[6] S. Amari, N. Mürata, K. -R, Muller, M. Finke, and H. Yang, "Asymptotic Statistical Theory of Overtraining and Cross-Validation," IEEE Trans. on Neural Networks, Vol.8, No.5, pp.985-996, 1997.

[7] E. B. Baum and D. Haussler, "What Size Net. Gives Valid Generalization," Neural Computation, Vol.1, pp.151-160, 1989.

[8] S. Bös, "How to Partition Examples between Cross- Validation Set, and Training Set ?," Lab. for Information Representation, RIKEN, 1996.

[9] A. R. Barron, "Approximation and Estimation Bounds for Artificial Neural Networks," Proceedings of the Fourth Annual Workshop on Computational Learning Theory, pp.243-249, 1991.

[10] A. R. Barron, "Complexity Regularization with Application to Neural Networks," Nonparametric Foundation Estimation and Related Topics, Roussas G., Editor, Kluwer Archademic Publishers, pp.561-576, 1991.

[11] hite, "Connectionist Nonparametric Regression : Multilayer Feedforward Networks Can Learn Arbitray Mappings," Neural Networks, Vol.3, pp.525-549, 1990.

[12] D. Barber, D. Saad, and P. Sollich, "Test Error Fluctuations in Finite Linear Perceptrons," Neural Computation, Vol.7, No.4, pp.80-821, 1995.

[13] S. Haykin, "Neural Networks : A Comprehensive Foundation," Macmillan Publishing Company, 1994.

[14] M. P. Lyu, "Handbook of Software Reliability Engineering," McGraw-Hill, 1995.

[15] N. Karunanithi, D. Whitley and Y. K. Malaiya, "Prediction of Software Reliability Using Connectionist Models," IEEE Trans. Software Eng., Vol.18, pp.563-574, 1992.

[16] N. Karunanithi, D. Whitley and Y. K. Malaiya, "Using Neural Networks in Reliability Prediction," IEEE Software, Vol.18, pp.53-59, 1992.



**이 상 운**

e-mail : sangun_lee@hanmail.net

1983년~1987년 한국항공대학교 항공전자 공학과(학사)

1995년~1997년 경상대학교 컴퓨터 과학과(석사)

1998년~2001년 경상대학교 컴퓨터 과학과(박사)

1992년~현재 국방품질관리소 항공전자장비 및 소프트웨어 품질보증 담당

관심분야 : 소프트웨어 공학(소프트웨어 시험 및 품질보증, 소프트웨어 신뢰성), 소프트웨어 메트릭스, 신경망, 뉴로-퍼지